Multi-Root I/O Virtualization and Sharing
Revision 0.5

November 13, 2006

REVISION	REVISION HISTORY	DATE
0.5	Initial release	10/30/2006

PCI-SIG disclaims all warranties and liability for the use of this document and the information contained herein and assumes no responsibility for any errors that may appear in this document, nor does PCI-SIG make a commitment to update the information contained herein.

Contact the PCI-SIG office to obtain the latest revision of the specification.

Questions regarding this document or membership in PCI-SIG may be forwarded to:

Membership Services
http://www.pcisig.com
E-mail: administration@pcisig.com
Phone: 503-291-2569
Fax: 503-297-1090

Technical Support
Technical support for this specification is available to members. For information, please visit: http://www.pcisig.com/developers/technical_support.

DISCLAIMER

This document is provided "as is" with no warranties whatsoever, including any warranty of merchantability, non-infringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. PCI-SIG disclaims all liability for infringement of proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

PCI Express, PCIe, PCI-X, and PCI-SIG are trademarks of PCI-SIG.

All other product names are trademarks, registered trademarks, or service marks of their respective owners.

Copyright © 2006 PCI-SIG
All rights reserved.

Contents

1. ARCHITECTURAL OVERVIEW 13

1.1. How does MR-IOV Work? 15

1.1.1. MRA Components 19

1.1.1.1. Multi-Root Aware Root Port (MRA RP) 19

1.1.1.2. Multi-Root Aware PCIe Device (MRA PCIe Device) 20

1.1.1.3. Multi-Root PCI Manager (MR-PCIM) 21

1.1.1.4. Multi-Root Aware PCIe Switch (MRA PCIe Switch) 23

1.1.2. MR Initialization Overview 24

1.1.3. MR Transaction Encapsulation Overview 24

1.1.4. MR Congestion Management Overview 24

1.1.5. MR Error and Event Handling Overview 24

1.1.6. MR-IOV and ARI (Alternative Routing Identifier) 24

1.1.7. MR-IOV Relationship to SR-IOV and ATS 24

2. MR PROTOCOL 25

2.1. Overview 25

2.1.1. Protocol Goals 25

2.2. Protocol Requirements 25

2.2.1. MR Protocol Changes 26

2.3. Transaction Layer 26

2.3.1. Overview of MR TL 26

2.3.2. TLP Prefix Header 30

2.3.3. MR Switch TL Processing 32

2.3.4. MR TLP Dataflow Examples 33

2.3.5. Interrupts 34

2.3.5.1. INTx Device Processing 34

2.3.5.2. INTx Switch Processing 35

2.3.5.3. INTx Root Port Processing 35

2.3.6. ARI Support 35

2.4. Per-VH RESET 35

2.4.1. Per-VH Reset Example 36

2.4.2. RESET DLLP Format 38

2.4.3. RESET DLLP Processing 39

2.4.3.1. Upstream State Machine 39

2.4.3.2. Downstream State Machine 41

2.4.3.3. Flow Control and Reset / DL_DOWN 42

3. INITIALIZATION AND RESOURCE ALLOCATION 44

3.1. Requirements 44

3.2. MR Topology Initialization 44

3.2.1. Fundamental Reset Initialization 45

3.2.1.1. PCIM Capable Ports 45

3.2.1.2. Non-PCIM Capable Ports 46

3.2.1.3. Non-PCIe Management Ports 46
3.2.1.4. Initial Port Type Example 48
3.2.2. MR-PCIM Topology Discovery 49
3.2.2.1. MR-PCIM Topology Discovery Example 50
3.2.3. MR-PCIM Failover 52
3.3. MR DEVICE INITIALIZATION 52
3.3.1. Enabling MR Operation 53
3.3.2. Managing Flow Control 54
3.3.3. Managing VF Mapping 55
3.3.4. Managing VF Migration 57
4. CONFIGURATION 60
4.1. REQUIREMENTS 60
4.2. CONFIGURATION FIELD SUMMARY 61
4.3. DEVICE CONFIGURATION SPACE 64
4.3.1. Device MR-IOV Extended Capability 65
4.3.1.1. PCI Express Extended Capability ID (00h) 66
4.3.1.2. Capability Version (02h) 66
4.3.1.3. Next Capability Offset (02h) 66
4.3.1.4. MR-IOV Capabilities (04h) 67
4.3.1.5. MR IOV Control (08h) 67
4.3.1.6. MR-IOV Status (0Ch) 67
4.3.1.7. MaxVH (10h) 68
4.3.1.8. NumVH (12h) 68
4.3.1.9. Management VH (14h) 68
4.3.1.10. VH Authorization Bitmap Offset (17h) 68
4.3.1.11. Memory Mapped Configuration Space Offset (18h) 69
4.3.2. Memory Mapped Configuration Space 69
4.3.2.1. MR Max MVFs 71
4.3.2.2. MR MVF Table Size 71
4.3.2.3. PF_N Max LVFs 71
4.3.2.4. PF_N Num LVFs 71
4.3.2.5. PF_N VF State Array 71
4.3.2.6. The MVF Table 72
4.3.3. The VF States (or MR-PCIM IOV Resource Initialization) 72
4.3.4. Capabilities (physical device stuff, etc) 73
4.3.5. Virtual Power Management Controller 73
4.3.6. Actual Power Management Controller 74
4.4. SWITCH CONFIGURATION SPACE 74
4.4.1. Switch MR-IOV Extended Capability 76
4.4.1.1. Switch MR-IOV Extended Capability Header (00h) 76
4.4.1.2. Switch MR-IOV Capability (04h) 76
4.4.1.3. Switch MR-IOV Control (08h) 76
4.4.1.4. Switch MR-IOV Status (0Ch) 77
4.4.1.5. MaxVH (10h) 77
4.4.1.6. NumVH (12h) 77
4.4.1.7. Management VS (14h) 77

4.4.1.8. VS Authorization Bitmap Offset (17h) 77
4.4.1.9. Port# / VS # / VS Bridge# (18h) 77
4.4.1.10. Physical Port Entry Size / Num Physical Port Entries (1Ch) 78
4.4.1.11. Physical Port Entry Table Offset (20h) 78
4.4.1.12. VS Table Entry Size / Num VS Table Entries (24h) 78
4.4.1.13. VS Table Offset (28h) 78
4.4.1.14. VS Bridge Table Entry Size / Num VS Bridge Table Entries per VS (2Ch) 78
4.4.1.15. VS Bridge Table Offset (30h) 79
4.4.2. Switch Physical Port Table 79
4.4.2.1. Port Present (RO) 79
4.4.2.2. Non-PCIe Port (RO) 80
4.4.2.3. Port Enable (RW) 80
4.4.2.4. MR-IOV Enable (RW) 80
4.4.2.5. Link Direction Status (RO) 80
4.4.2.6. Link Partner Detected (RO) 80
4.4.2.7. Link Direction Control (RW) 80
4.4.2.8. Total VH Implemented (RO) 80
4.4.2.9. MaxVH (RW) 80
4.4.2.10. NumVH (RO) 80
4.4.2.11. VL Arbitration fields 81
4.4.3. Switch VS Table 81
4.4.3.1. VS Present 81
4.4.3.2. VS Enable 81
4.4.3.3. VS Interrupt Enable 82
4.4.3.4. VS Global Key Check Enable 82
4.4.3.5. VS Global Key Value 82
4.4.3.6. $Bridge_N$ Interrupt Status 82
4.4.4. Switch VS Bridge Table 83
4.4.4.1. Bridge Present (RO) 84
4.4.4.2. Bridge Enable (RW) 84
4.4.4.3. Num VC Resources Hardware Present (RO) 84
4.4.4.4. Extended VC Count (RW) 84
4.4.4.5. Low Priority Extended VC Count (RW) 85
4.4.4.6. Slot Power Limit Value (RO) 85
4.4.4.7. Slot Power Limit Scale (RO) 85
4.4.4.8. Bridge VHN (RW) 85
4.4.4.9. Bridge Port (RW) 85
4.4.4.10. VCn to VL Map (RW) 85
4.4.4.11. VC Resource n fields (RO) 86
4.4.4.12. Virtual Hot-Plug Control Bits 86
5. ERROR HANDLING 89
5.1. REQUIREMENTS 89
5.2. PCIE ERROR MAPPING TO MR 89
5.3. MR ERRORS 92

6. HOT PLUG 93
6.1. REQUIREMENTS 93
6.2. MRA SWITCH 93
6.2.1. Virtual Slot Registers 93
6.2.2. Virtual Signals Interface Registers 96
6.2.3. Physical Slot Registers 97
6.2.4. Physical Signals Interface 98
6.3. MRA DEVICE 98
6.4. NON-MRA COMPONENTS 98
7. POWER MANAGEMENT 99
7.1. REQUIREMENTS 99
7.2. VIRTUAL POWER MANAGEMENT 99
7.2.1. Virtual Wake 99
7.3. ACTUAL POWER MANAGEMENT 100
7.3.1. Actual Wake 100
8. CONGESTION MANAGEMENT 101
8.1. OVERVIEW 101
8.2. REQUIREMENTS AND GOALS 101
8.3. VIRTUAL LINKS WITH BYPASS QUEUES 102
8.3.1. (VH, VC) Mapping 104
8.3.2. Arbitration 104
8.3.2.1. VH to VL Arbitration 105
8.3.2.2. VL to Link Arbitration 105
8.4. MR FLOW CONTROL 106
8.4.1. SR to MR Enabled Link Transition 106
8.4.2. FC Information Tracked by Transmitter 106
8.4.3. FC Information Tracked by Receiver 107
8.4.4. Virtual Hierarchy (VH) Reset 107
8.4.5. Enabling and Disabling of VCs 108
8.5. PERFORMANCE MONITORING 108
8.5.1. Statistics Collections 108
8.5.2. Counter Values 109
8.5.3. Sampled Values 110

Figures

Figure 1-1: Generic Server Blade Configuration 13
Figure 2-1: Example Multi-Root Topology 27
Figure 2-2: Example Multi-Root Topology as viewed from Host A 29
Figure 2-3: Example Multi-Root Topology as viewed from Host C 30
Figure 2-4: TLP Prefix Header Location 31
Figure 2-5: TLP Prefix Layout (tentative) 31
Figure 2-6: MR Dataflow Examples 33

Figure 2-7: Reset DLLP Example: Topology....36
Figure 2-8: Reset DLLP Example: Host A's View....37
Figure 2-9: Reset DLLP....39
Figure 2-10: Upstream Link Partner RESET SM....41
Figure 2-11: Downstream Link Partner RESET SM....42
Figure 3-1: Example MR Topology....48
Figure 3-2: Example MR Topology with Initial Link Directions....50
Figure 3-3: Example Mapping of VFs associated with PF 0....56
Figure 3-4: VF Migration State Diagram....58
Figure 4-1: MR Device Configuration Space....65
Figure 4-2: Device MR-IOV Capability....66
Figure 4-3: MR Device PF Table....69
Figure 4-4: MR Device Global Key Table....70
Figure 4-5: MR Device PF / VH Table....70
Figure 4-6: Multi-Root IO Virtualization Memory Mapped Configuration Space....71
Figure 4-7: VF State Table Diagram....73
Figure 4-8: Switch Mapping Tables....75
Figure 4-9: Switch MR-IOV Capability....76
Figure 4-10: Switch Physical Port Table....79
Figure 4-11: VS Table....81
Figure 4-12: VS Bridge Table....83
Figure 6-1: Slot Capabilities Register (PCIe Figure 7-18)....94
Figure 6-2: Slot Control Register (PCIe Figure 7-19)....95
Figure 6-3: Slot Status Register (PCIe Figure 7-20)....96
Figure 8-1: (VH, VC) to VL Mapping....103
Figure 8-2: Logical Queuing Structure Associated with a VL Receiver....103
Figure 8-3: MRA Arbitration Model....105
Figure 8-4: Statistics Collection Process....108

Tables

Table 2-1: Reset DLLP Example: Events and Actions....37
Table 3-1: Port Types – Example MR Topology....49
Table 3-2: Valid MR State Transitions for VF Migration....58
Table 4-1: MR-IOV Fields....61
Table 4-2: MR-IOV Capabilities (04h)....67
Table 4-3: MR-IOV Control (08h)....67
Table 4-4: MR-IOV Status (0Ch)....68
Table 5-1: Physical Layer Error List....89
Table 5-2: Data Link Layer Error List....90
Table 5-3: Transaction Layer Error List....91
Table 5-4: MR Error List....92

Objective of the Specification

The purpose of this document is to specify PCI I/O virtualization and sharing technology. The specification is focused on multi-root topologies, e.g. a server blade enclosure that uses a PCI Express Switch-based topology to connect server blades to PCI Express Devices or PCI Express-to-PCI Bridges and enable the leaf Devices to be serially or simultaneously shared by one or more server blades.

This document is to be used in conjunction with, and does not supersede, the terms and conditions specified in the *PCI-SIG™ Trademark and Logo Usage Guidelines* document.

Document Organization

Chapter 1 specifies .

Chapter 2 specifies .

Documentation Conventions

Capitalization

Some terms are capitalized to distinguish their definition in the context of this document from their common English meaning. Words not capitalized have their common English meaning. When terms such as "memory write" or "memory read" appear completely in lower case, they include all transactions of that type.

Register names and the names of fields and bits in registers and headers are presented with the first letter capitalized and the remainder in lower case.

Numbers and Number Bases

Hexadecimal numbers are written with a lower case "h" suffix, e.g., 0FFFFh and 80h. Hexadecimal numbers larger than four digits are represented with a space dividing each group of four digits, as in 1E FFFF FFFFh. Binary numbers are written with a lower case "b" suffix, e.g., 1001b and 10b. Binary numbers larger than four digits are written with a space dividing each group of four digits, as in 1000 0101 0010b.

All other numbers are decimal.

Reference Information

Reference information is provided in various places to assist the reader and does not represent a requirement of this document. Such references are indicated by the abbreviation "(ref)." For example, in some places, a clock that is specified to have a minimum period of 400 ps also includes the reference information maximum clock frequency of "2.5 GHz (ref)." Requirements of other specifications also appear in various places throughout this document and are marked as reference information. Every effort has been made to guarantee that this information accurately reflects the referenced document; however, in case of a discrepancy, the original document takes precedence.

Implementation Notes

Implementation Notes should not be considered to be part of this specification. They are included for clarification and illustration only. Implementation Notes within this document are enclosed in a box and set apart from other text.

Terms and Abbreviations

Base Function (BF)	Function used to manage the MR features of an MR Device.
IO Virtualization (IOV)	The capability for a single PCIe component to be used by more than one SI.
Hypervisor (HV)	A software component supporting one or more SIs. Specification of the behavior and architecture of HV is outside the scope of this specification.
Management Port	A connection to a MR Switch that can be used to manage a MR topology. Management Ports may be actual PCIe ports or may be Vendor Specific ports non-PCIe ports.
MR - Multi-Root Topology	A PCIe topology that interconnects more than one Root Port through multi-root aware Switches.
MR PCIM	Multi-Root PCIM
MRA	Multi-Root Aware – A PCIe component that supports the multi-root extensions defined in this specification.
MRA VF	Virtual Function (VF) in an MRA Device
MR Enabled Link	PCIe link using the Multi-Root Encapsulated Link Protocol.
MR Egress	Point in a PCIe fabric that a TLP exits the MR Fabric
MR Fabric	Subset of a PCIe fabric containing MR Enabled Links and connected MRA components.
MR Ingress	Point in a PCIe fabric that a TLP enters the MR Fabric.
PCI Bus Memory Address	Refers to the address portion of a PCI Memory Transaction
PCI Manager (PCIM)	Software that enumerates and configures an IOV topology.
PCIe	PCI Express
Physical Address	The address used by the system memory controller to access system memory.
Physical Function (PF)	An IOV-capable Function per the *Single Root IOV Specification*. In MR, PFs exist within Virtual Hierarchies (VHs).
RC	Root Complex per the *PCI Express Base Specification*
RP	Root Port per the *PCI Express Base Specification*
System Image (SI)	A software component running on a Virtual System to which specific

	virtual and physical Devices can be assigned. Specification of the behavior and architecture of an SI is outside the scope of this specification. Examples of SIs include guest operating systems and shared/non-shared protected domain Device drivers.
SR PCIM	Single Root PCIM
Virtual Device	Collection of PFs and VFs that operate as a Device within a VH.
Virtual Function (VF)	An IOV-capable Function per the *Single Root IOV Specification*. In MR, VFs exist within Virtual Hierarchies (VHs).
Virtual Hierarchy (VH)	Portion of an MR Topology assigned to a single PCIe Domain Hierarchy.
Virtual Hierarchy Number (VHN)	Link Local Number designating a VH.
Virtual Intermediary (VI)	A software component supporting one or more SIs – colloquially known as a hypervisor or virtual machine monitor. Specification of the behavior and architecture of VI is outside the scope of this specification.
Virtual Switch (VS)	A logical PCIe Switch associated with a single VH implemented in a MR Switch.

1

1. Architectural Overview

Within the industry, significant effort has been expended to increase the effective hardware resource utilization through the use of virtualization technology. The Multi-Root I/O Virtualization (MR-IOV) specification defines the extensions to the PCI Express (PCIe) specification suite to enable multiple non-coherent Root Complex (RC) to share PCI hardware resources.

To illustrate how this technology can be used to increase effective resource utilization, consider the following generic server blade configuration as illustrated in the Figure 1-1 below.

- The server blade configuration contains four server blades and two external fabric switches. In a high availability configuration, nominally there would be two external fabric switches of each type to avoid any single point of failure (SPOF) for a total of four switches.
- In this example, each switch provides two external connectivity ports though more can be configured to deliver high availability solutions as well as increased aggregate performance.
- Each server blade is provisioned with two PCIe Endpoint Devices – an Ethernet and a storage area network (SAN) device. This translates to a total of eight PCIe Endpoint Devices. These PCIe Devices are point-to-point connected to a Root Port (RP) [not shown] – either emitted by a chipset or a processor.
- Each I/O device and switch port is typically provisioned to enable any I/O device to operate at full bandwidth.

Depending upon workload, the example configuration's I/O resource capacity may be excessive resulting in under-utilized hardware.

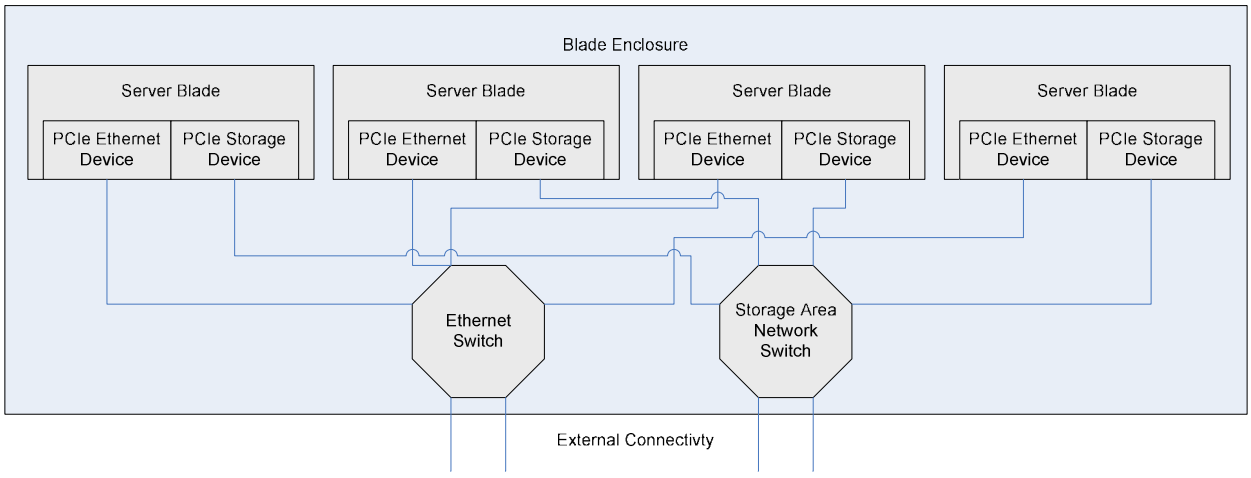


Figure 1-1: Generic Server Blade Configuration

Through the application of MR-IOV technology, the prior example server blade configuration can be transformed as illustrated in Figure 1-2.

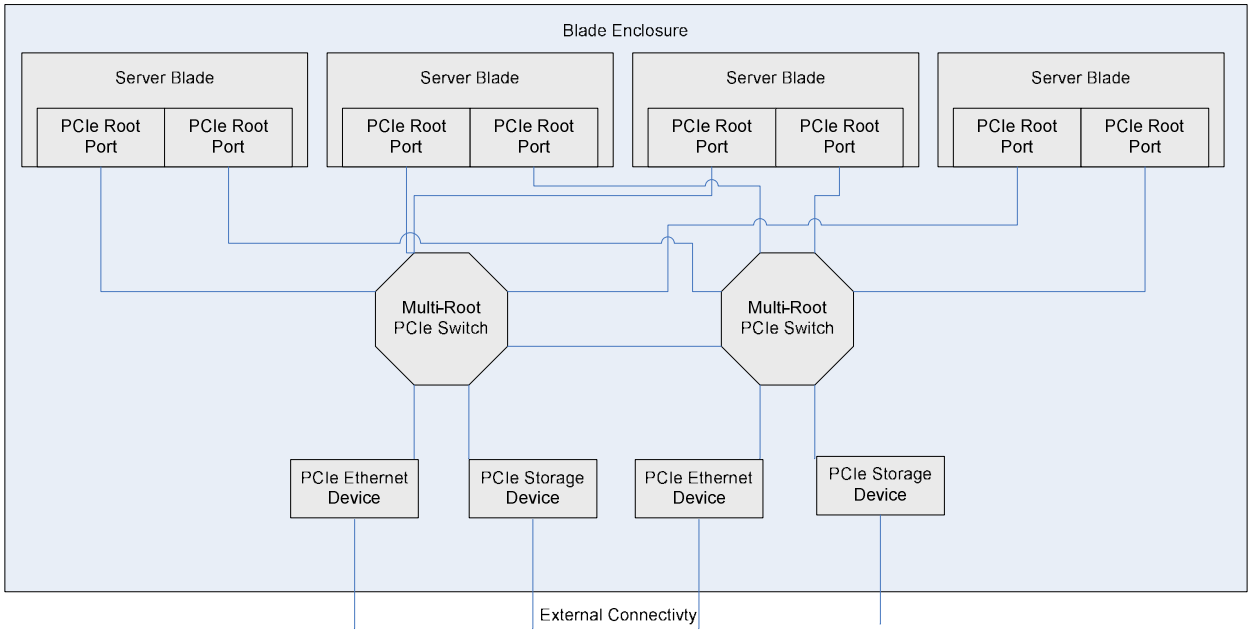


Figure 1-2: Example Server Blade Configuration using MR-IOV Technology

In contrast to the figure 1-1, the following can be observed:

- The server blade configuration contains four server blades. The server blades do not contain PCIe Endpoint Devices but instead connect a Root Port (RP) to a Multi-Root Aware (MRA) PCIe Switch.
- The two external fabric switches are replaced by two MRA PCIe Switches. While the details will be described in a subsequent section, the following should be noted:
 - Unlike a PCIe Switch which contains a single upstream Port and can only be claimed by a single RP, a MRA PCIe Switch contains multiple upstream Ports to enable it to connect to multiple RP. This enables the MRA PCIe Switch to be a shared component within the configuration.
 - Multiple MRA PCIe Switches can be interconnected in a variety of topologies to create high availability solutions as well as provide increased I/O fan-out capacity.
- In place of eight PCIe Endpoint Devices – four of each type – the example MR-IOV configuration contains four MRA PCIe Endpoint Devices – two of each type. Each MRA PCIe Endpoint Device is attached to a MRA PCIe Switch downstream Port enabling each to be accessed, and thus shared, by any of the server blades.
- Unlike the prior example configuration where I/O is dedicated to each server blade, a MR-IOV based configuration enables the I/O to be dynamically assigned. A fraction or an entire I/O Device can be assigned to each server blade based on its workload requirements.

As noted above, a MR-IOV configuration reduces the component count and change the component composition. This specification will cover the elements involved in delivering a MR-IOV configuration.

1.1. How does MR-IOV Work?

To understand how MR-IOV works, first examine an example platform configuration devoid of any Single Root IOV (SR-IOV) – see the *Single-Root I/O Virtualization Specification* - or MR-IOV technology as illustrated in Figure 1-3.

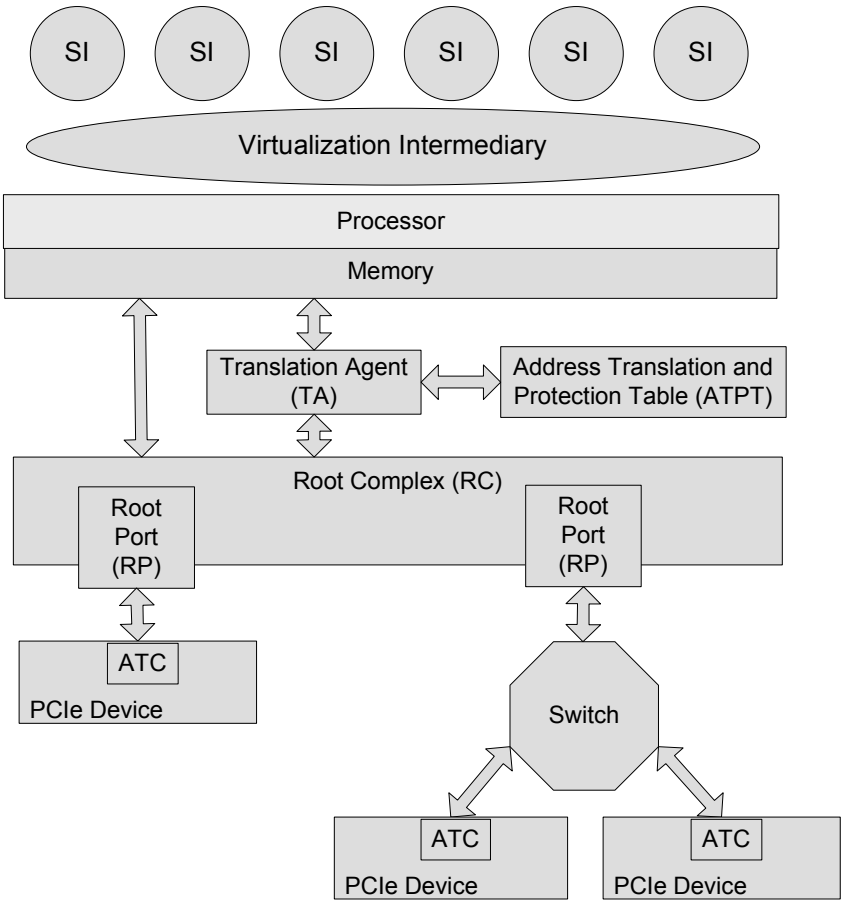


Figure 1-3: Example Platform Configuration without SR-IOV or MR-IOV Technology

The above example platform is composed of the following:

- A processor capable of running any operating system (OS). In virtualized environments, a processor can execute a Virtual Intermediary (VI) which abstracts or virtualizes all hardware from one or more System Images (SI). Each SI contains a virtual I/O device driver while the VI will contain the I/O-specific device drivers and perform all I/O hardware accesses.
- A memory controller and associated memory.
- A Translation Agent (TA) and Address Translation and Protection Table (ATPT).
 - A TA parses the contents of a PCIe DMA request transaction (TLP) to index an ATPT to derive the physical address translation and access rights. The purposes for having DMA address translation vary and include:

- Limiting the destructiveness of a 'broken' or miss-programmed DMA I/O Function
- Providing for scatter/gather
- Ability to redirect message-signaled interrupts (e.g., MSI or MSI-X) to different address ranges without requiring coordination with the underlying I/O Function
- Address space conversion (32-bit I/O Function to larger system address space)
- Virtualization support

 - A PCIe Endpoint may contain an Address Translation Cache (ATC) in support of the PCI-SIG *Address Translation Services Specification*.

- A PCIe Root Complex (RC) containing one or more Root Ports (RP) with direct-attached or PCIe Switch-attached PCIe Devices or PCI / PCI-X Bridges. Each RP defines a unique hierarchy domain (see *PCI Express Base Specification*).

Now examine a platform that supports SR-IOV technology as illustrated in Figure 1-4 below.

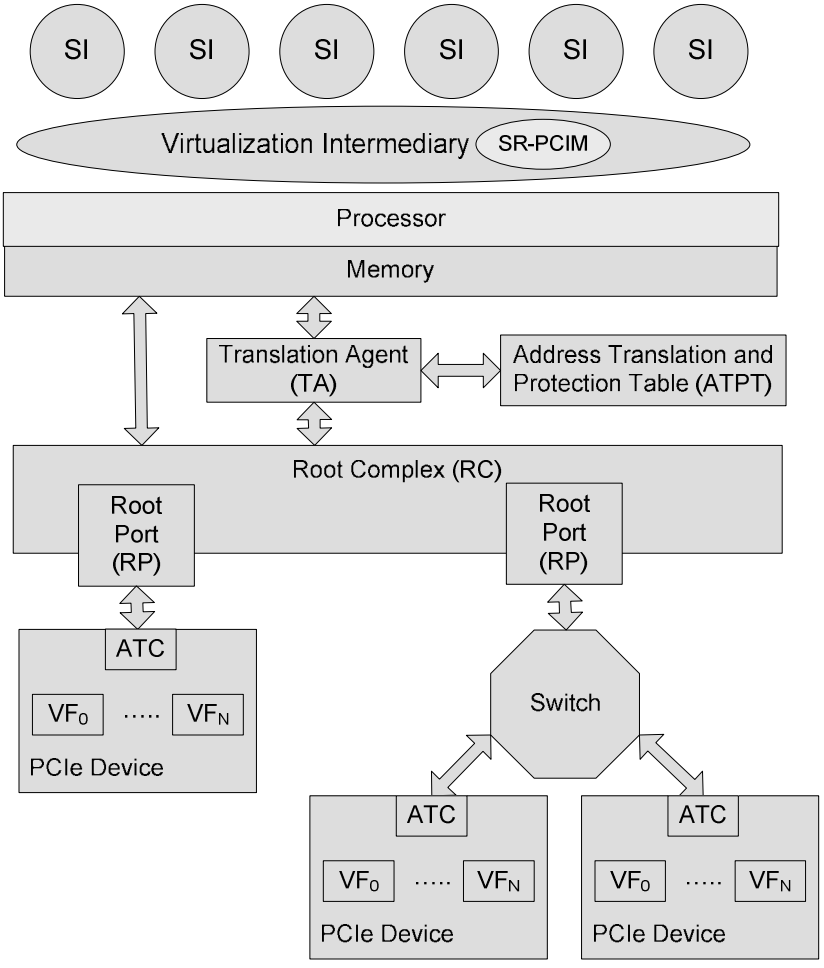


Figure 1-4: Example Platform Configuration with SR-IOV Technology

The differences between the platforms in Figure 1-3 and Figure 1-4 are:

- The PCIe Device support the SR-IOV capability as defined in the *Single Root I/O Virtualization Specification*. SR-IOV enables a PCIe Device to support multiple Virtual Functions (VF).
- SR-PCIM (Single Root PCI Manager) is responsible for SR-IOV hardware configuration and management. SR-PCIM is nominally integrated within a VI, though a variety of implementation options exist and remain outside the scope of the PCI-SIG specifications.

From the prior two figures, the following key semantics should be kept in mind:

1. The OS or VI has exclusive control over each PCIe Component – RC, RP, Switch, Link, PCIe-to-PCI/PCI-X Bridge, Device, and Function.
 a. All PCI enumeration, configuration operations, reset, power management, and event handling, e.g. errors, must only be initiated or processed by the OS or VI. Operations initiated by a SI are trapped by the VI and processed on its behalf.
2. Each RP acts as the terminus for all upstream targeted operations, e.g. error event notification.

In order to support either example platform while preserving these semantics, the PCI components underneath each RP must be virtualized and logically overlaid on the MRA PCIe Switches and Devices as illustrated in Figure 1-5. The virtualized PCI components are referred to as a Virtual Hierarchy (VH). A VH has the following attributes:

- Each VH must contain at least one PCIe Switch.
 - The PCIe Switch will be a virtualized component implemented over of a MRA Switch.
 - The PCIe Switch functionality and semantics are per the *PCI Express Base Specification*.
- Each VH may contain any mix of PCIe Devices, MRA PCIe Devices, or PCIe to PCI / PCI-X Bridges as illustrated in Figure 1-6.
 - A PCIe Device is a device that does not support the MR-IOV Capability. Such a device must only be visible in a single VH at a time. A PCIe Device can be serially shared among a set of accessible VH within the MR-IOV topology. The PCIe Device is serially deleted from the current source VH and added to destination VH.
 - A PCIe Device may support the SR-IOV Capability which enables it to be shared by multiple SI executing above a single RP.
 - A PCIe to PCI / PCI-X Bridge can only be visible in a single VH at a time. As with a PCIe Device, a PCIe to PCI / PCI-X Bridge can be serially shared among a set of accessible VH within the MR-IOV topology using a conceptually similar deletion / addition process as a PCIe Device.
 - The SR-IOV Capability does not apply to the PCIe to PCI / PCI-X Bridge. The bridge and all associated PCI / PCI-X devices can only be configured in a single OS, VI, or SI at a time.
 - A MRA PCIe Device is a device which supports the MR-IOV Capability. Such a device can be visible in multiple VH at a time depending upon the MR-IOV

resources provisioned. A MRA PCIe Device can be added or deleted from any accessible VH within a MR-IOV topology.

- A MRA PCIe Device must support the SR-IOV Capability. This enables it to be shared by multiple SI executing above each RP.

- The MR-IOV topology must contain at least one MRA PCIe Switch.
 - Multiple MRA PCIe Switches can be provisioned and interconnected in a variety of topologies – tree, fat-tree, star, mesh, etc.
 - A MRA PCIe Switch must contain two or more upstream Ports. Each upstream Port must connect to a RP which acts as the root of the VH.

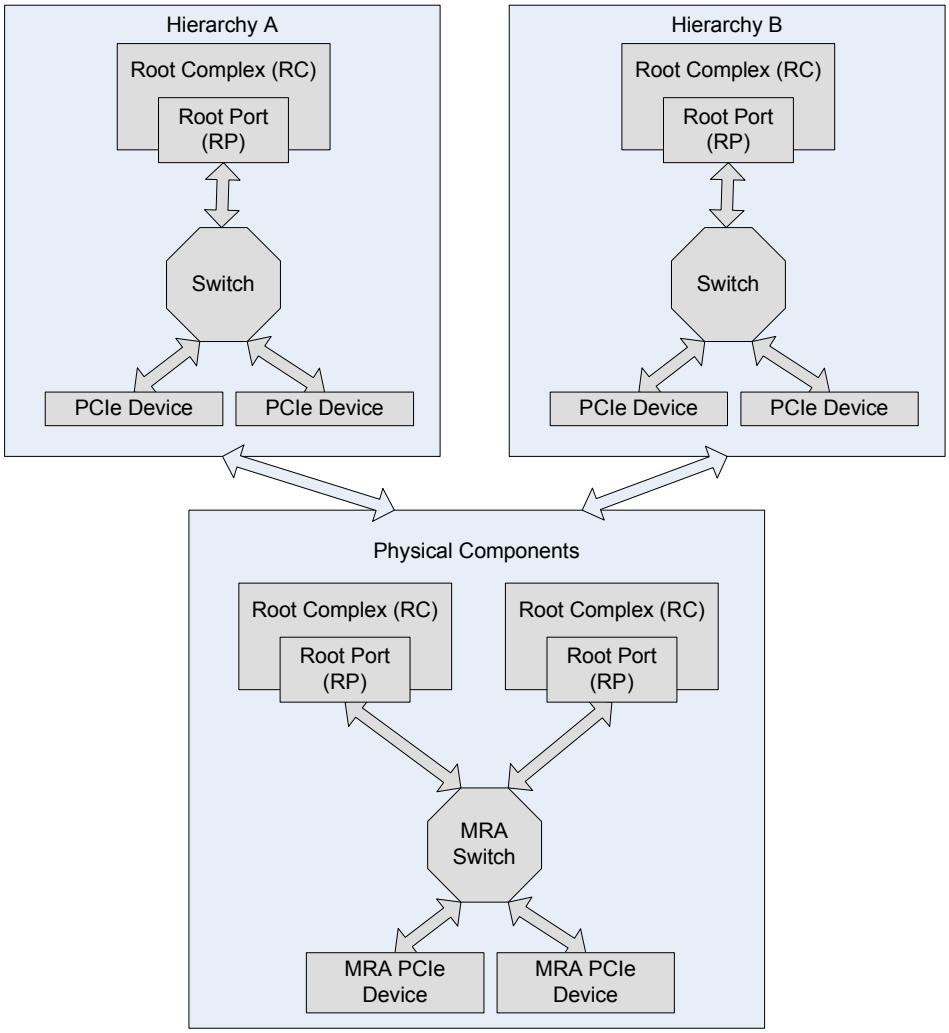


Figure 1-5: Two Virtual Hierarchies (VH) Implemented over Shared Physical Components

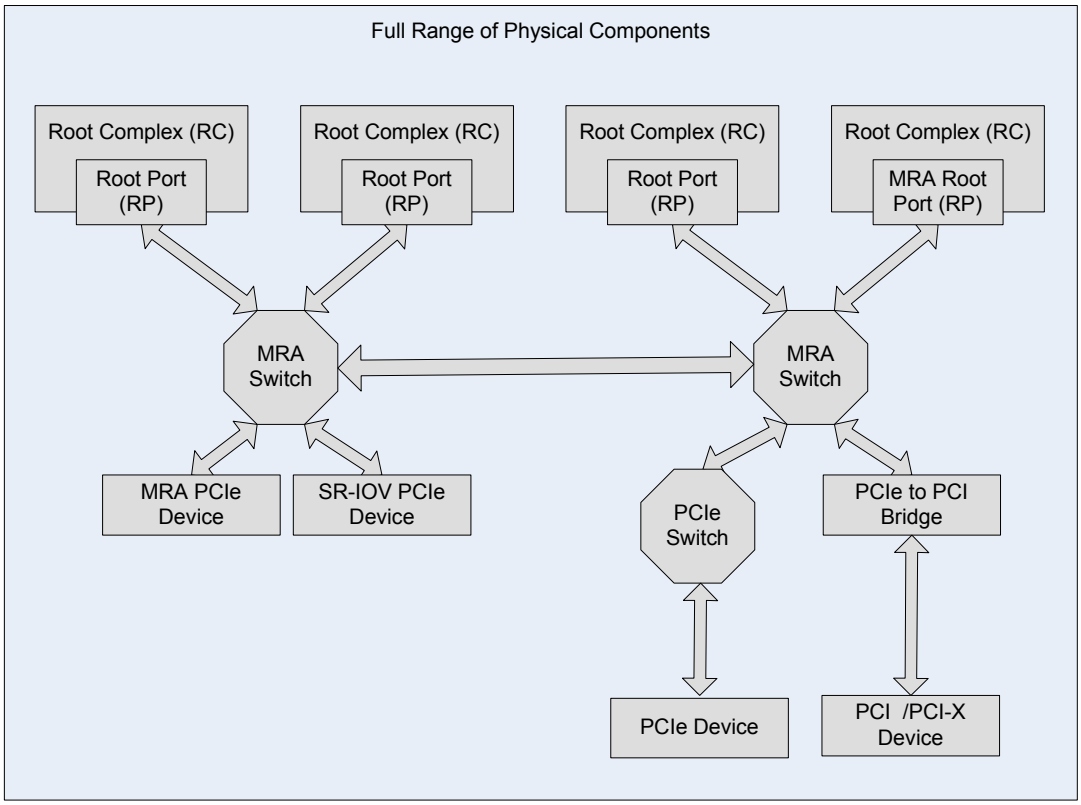


Figure 1-6: Physical Components that can be supported in a MR-IOV Topology

1.1.1. MRA Components

The prior section illustrated that MR-IOV is primarily the overlaying of multiple VH over a shared physical set of MRA and non-MRA components. To further understand how MR-IOV works, let's examine the MRA components in more detail.

1.1.1.1. Multi-Root Aware Root Port (MRA RP)

As illustrated in Figure 1-6, a PCIe RP supporting either a single OS or a VI with multiple SI can be connected to a MRA Switch and access multiple downstream devices and bridges. The PCIe RP though, is restricted to a single VH. In order to enable multiple VH to be accessed, a MRA PCIe RP is required. A MRA PCIe RP differs from a PCIe RP in the following ways:

- A MRA PCIe RP maintains state to delineate each VH. At a high level, this amounts to a set of resource mapping tables to translate the I/O function associated with each SI into a VH and MR I/O function identifier.
- A MRA PCIe RP participates in the MR transaction encapsulation protocol (see subsequent section for details) to enable a MRA PCIe Switch to derive the VH and associated routing information.

- A MRA PCIe RP emits a MRA Link. A MRA Link is identical the physical layer to a PCIe Link as defined in the *PCI Express Base Specification*. A MRA Link differs at the data link layer where a new set of DLLP are defined to support the MR-IOV protocol.
- A MRA PCIe RP may implement MRA congestion management (see subsequent chapter for details).

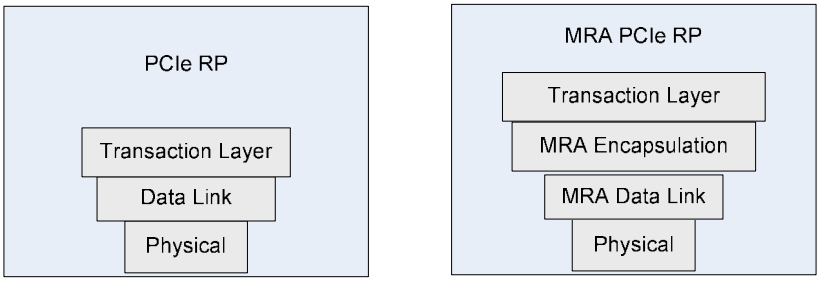


Figure 1-7 PCIe RP and MRA PCIe RP Functional Block Comparison

1.1.1.2. Multi-Root Aware PCIe Device (MRA PCIe Device)

A MR-IOV platform may contain any mix of PCIe Devices, SR-IOV PCIe Devices, or MR-IOV PCIe Devices. Figure 1-8 illustrates a functional block comparison between these three types of devices.

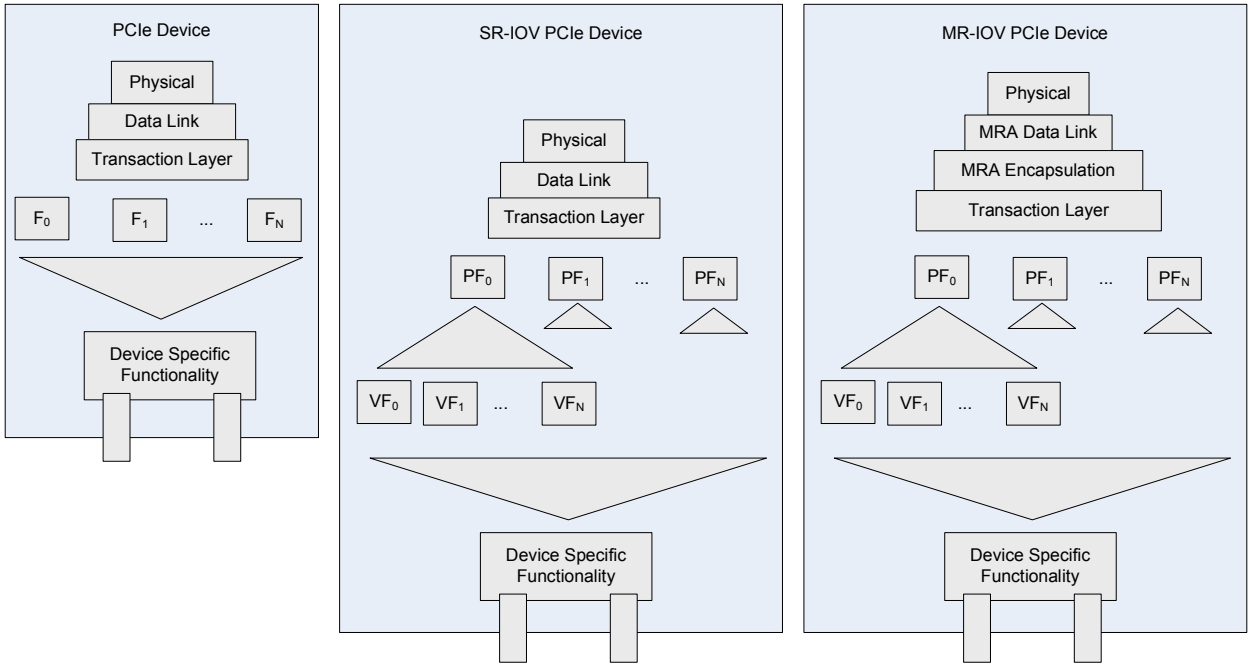


Figure 1-8 PCIe Device, SR-IOV, and MRA PCIe Device Functional Block Comparison

A MRA IOV PCIe Device differs from a PCIe Device and a SR-IOV PCIe Device in the following ways:

- The MRA IOV PCIe Device must support the new MR DLLP protocol.

 - A PCIe Device or a SR-IOV PCIe Device do not support the MR-IOV capability and therefore are unable to participate in this protocol. A MRA PCIe Switch must subsume all responsibility for forwarding transactions and event handling on behalf of these devices through the MR-IOV topology. The MRA PCIe Switch will perform all encapsulation or de-encapsulation as appropriate.
- The MRA IOV PCIe Device must support the MR-IOV transaction encapsulation protocol.
 - The MR-IOV encapsulation protocol provides VH identification information to the MRA PCIe Switch to enable the transaction to be transparently forwarded through the MR-IOV topology without requiring modification to the *PCI Express Base Specification* TLP protocol or contents.
- The MRA IOC PCIe Device is composed of a set of Physical Functions (PF).
 - Each PF supports a full PCI Configuration and PCIe Extended Configuration Space.
 - Each PF supports a full BAR.
 - Each PF must only be assigned to a single accessible VH at a given time.
 - One or more PF may be assigned to any accessible VH.
 - A PF and its associated resources may be migrated from one VH to another.
 - Each PF may support zero or more Virtual Functions (VF).
 - VF share resources including some portions of the configuration space with the associated PF.
 - A VF exists only within the VH associated with the PF.
 - A MRA PCIe Device with multiple PF and zero VF per PF is conceptually equivalent to a single function PCIe Device configured per VH.
 - The number of VF provisioned per PF may vary on a per PF basis.
 - Each PF represents a single device-specific functionality, e.g. an Ethernet controller, a SATA controller, etc. Subsequently, each VF must represent the same device-specific functionality. This enables the existing device driver models to be supported.

1.1.1.3. Multi-Root PCI Manager (MR-PCIM)

Each MRA component must support a corresponding MR-IOV capability. This capability is accessed and configured by the Multi Root PCI Manager (MR-PCIM). MR-PCIM can be implemented anywhere within the MR-IOV topology – for example, above a RP as illustrated in Figure 1-9, or, for example, through a private interface provided by a MRA PCIe Switch.

MR-PCIM is responsibilities include:

- Enumeration of the physical components within the MR-IOV topology. MR-PCIM must determine what components are or are not MR-IOV capable, how components are interconnected, and what the PCIe and MR-IOV resources they provide.

- MR-PCIM configures the components and resources that comprise each VH. The policies to determine this are outside the scope of this specification.
- Given the physical hardware is shared among a set of VH, MR-PCIM configures all PCIe and MR-IOV attributes including: link signaling rate, VC arbitration, Port arbitration, Access Control Services (ACS), etc.
- Given the physical hardware is shared among a set of VH, MR-PCIM processes or controls various events, e.g. RESET, physical hardware failure, surprise add / remove, error handling, etc.
- <continue to build up list of responsibilities>

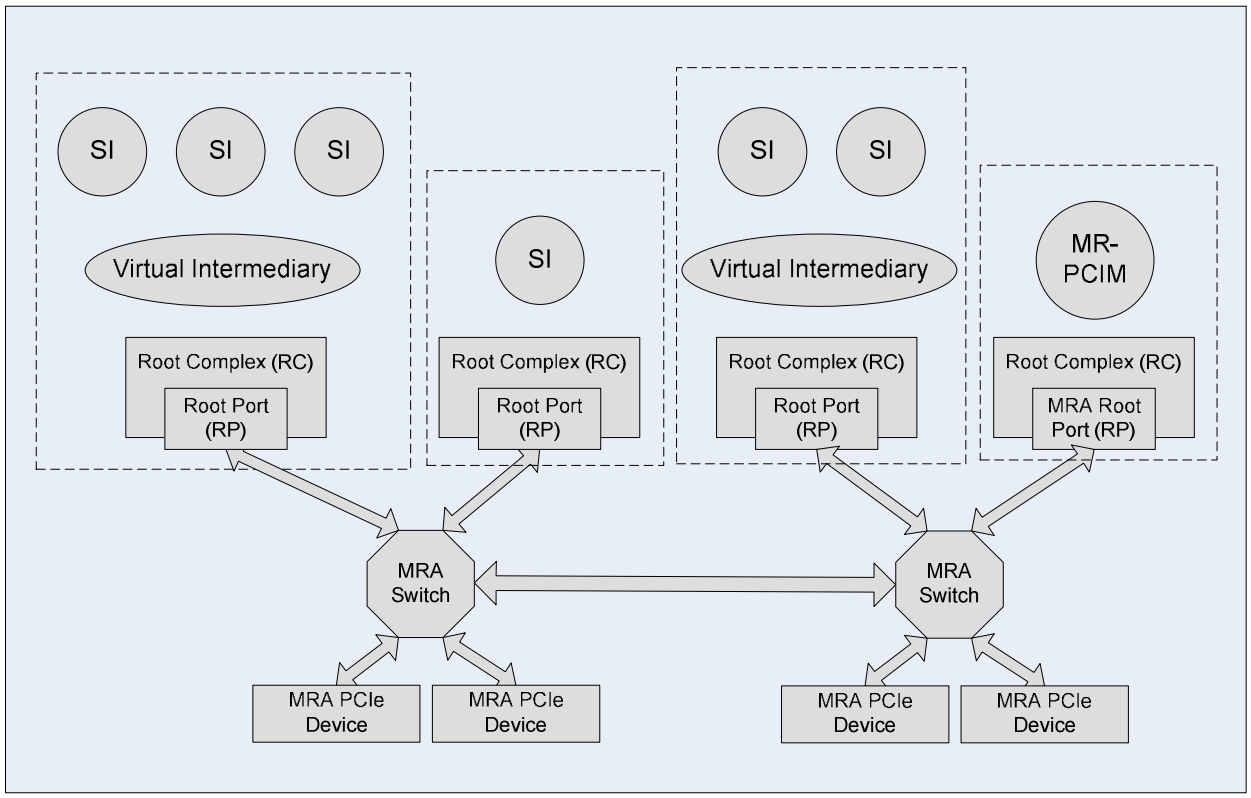


Figure 1-9 MRA PCIM in a MR-IOV Topology

1.1.1.4. Multi-Root Aware PCIe Switch (MRA PCIe Switch)

In a prior section it was noted that a MRA Switch is conceptually the overlay of multiple PCIe Switches onto a single physical package. This is illustrated in more detail in Figure 1-9.

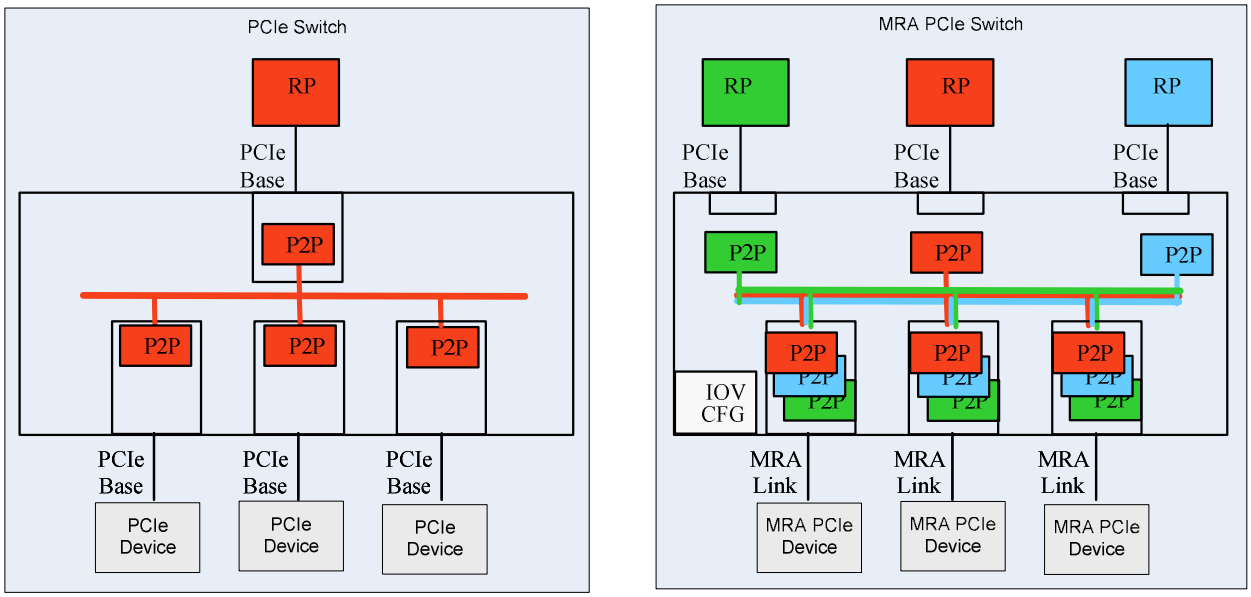


Figure 1-9 PCIe Switch and MRA PCIe Switch Functional Block Comparison

The PCIe Switch is composed of a set of logical P2P bridges with a single upstream Port attached to a PCIe RP and one or more downstream Ports attached to either a PCIe Device or a PCIe to PCI / PCI-X Bridge. A PCIe Switch also operates using a single address space.

In contrast a PCIe Switch, a MRA Switch is as follows:

- A MRA Switch is composed of one or more upstream Ports attached to either PCIe RP or MRA PCIe RP or the downstream Port of a MRA Switch.
 - If the upstream Port is attached to a PCIe RP, the MRA Switch must transparently provide all MRA related services on behalf of the PCIe RP.
- A MRA Switch is composed of one or more downstream Ports attached to either a PCIe Devices, a MRA PCIe Device, a PCIe Switch upstream Port, a MRA Switch upstream Port, or a PCIe to PCI / PCI-X Bridge.
- A set of logical P2P bridges that constitute a VH. A MRA Switch must support two or more VH.
- Each VH represent a separate address space. The combination of a VH identifier and the address contained within the PCIe TLP enable the MRA Switch to forward the a TLP to appropriate egress Port as well as a MRA RP or MRA PCIe Device to delineate which PF or VF is the source or sink of the PCIe TLP.
- <fill in additional attributes / operational semantics to flesh out this section>

1.1.2. MR Initialization Overview

<tbd>

1.1.3. MR Transaction Encapsulation Overview

<likely migrate sections within chapter 2 here>

1.1.4. MR Congestion Management Overview

<tbd>

1.1.5. MR Error and Event Handling Overview

<tbd>

1.1.6. MR-IOV and ARI (Alternative Routing Identifier)

<tbd>

1.1.7. MR-IOV Relationship to SR-IOV and ATS

Briefly:

- A MRA PCIe Device must support the SR-IOV Capability per the *Single Root I/O Virtualization Specification*.
 - A SR-IOV PCIe Device must support the PCI Express Base Specification.
 - By requiring these specifications to be supported, the number of permutations is reduced further enhancing the ability deploy and interoperate across a wide range of solution options.
- A MRA PCIe Device may support the *Address Translation Services Specification*.

2

2. MR Protocol

2.1. Overview

2.1.1. Protocol Goals

- DLL centered changes
- No changes to PHY Layer
- TL processing maintained within a VH
- Upward compatible with the *Single Root IOV Specification*
- Interoperates with PCIe components (especially Root Port and Devices)

2.2. Protocol Requirements

- The multi-root solution shall give each RP its own Virtual Hierarchy.
- The multi-root solution shall enable use of existing PCIe 1.1 or later RP.
- The multi-root solution shall enable existing PCIe 1.1 Switches, Devices, and PCIe to PCI/PCI-X Bridges to each be assigned to a single RP.
- The multi-root solution shall enable multiple MRA Switches to be connected such that multiple single rooted hierarchies can be interconnected, providing that the interconnected hierarchies appear to be a hierarchy structure to each RP.
- The multi-root solution shall not make special provisions to resolve loops created by connecting multiple MRA Switches together.
- The multi-root solution shall enable an RP to exist on any upstream Port of any MRA Switch.
- The multi-root solution should enable software written for PCI, PCI-X, or PCIe enumeration, configuration, and management to run, unchanged on each RP within the multi-root system
- Each Switch, Bridge, function, and VF shall be uniquely represented in the configuration space of each RP.

- Enabling RP-to-RP data transfer or other communications is a non-requirement for the multi-root solutions.
- Multi-root topologies can be built using single-root IOV Devices and RP. Non MRA components will be limited to a single virtual hierarchy. An MRA Switch is required to interconnect these components.
- Peer-to-peer traffic between PCI components in different virtual hierarchies shall not be allowed.
- The specification shall define the VHN as link local, at least 8 bits and protected end-end within the VH domain.
- The specification shall define the TLP prefix as being located between Sequence Number and PCIe TLP Header
- The specification shall not modify the TLP header while it traverses the MR topology.
- The specification shall support Per-VH Hot Reset implemented using DLLPs.
- Protection of "Switch VH Hopping" Requirement?

2.2.1. MR Protocol Changes

There are four parts of the PCIe Protocol that are changed to support Multi-Root operation.

- Tagging TLPs with a TLP Prefix Header
- Supporting per-VH equivalent of Hot Reset
- Configuration Space controls for managing and enabling MR operation
- Congestion Management Controls

2.3. Transaction Layer

2.3.1. Overview of MR TL

Figure 2-1 shows an example Multi-Root Topology

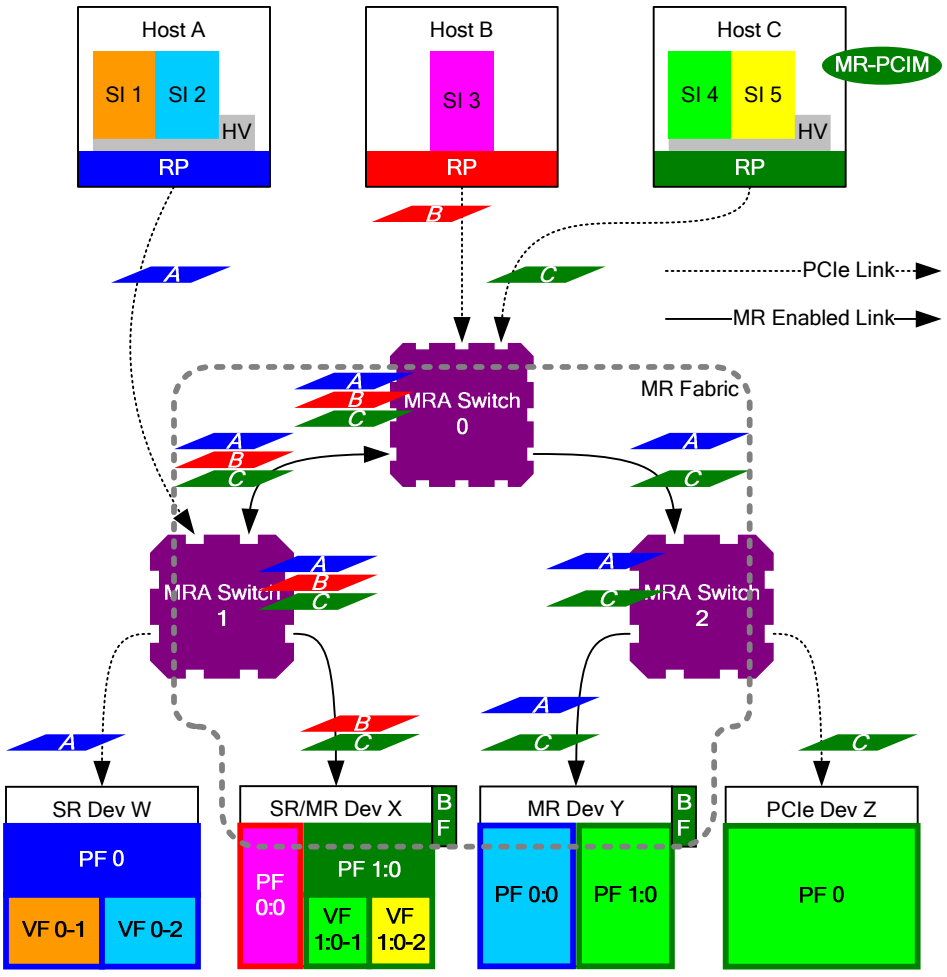


Figure 2-1: Example Multi-Root Topology

The solid links are using the Multi-Root Wire Protocol described in this specification. The dashed links are using the PCIe Protocol. SR Functions are designated PF *f* and VF *f-s* where *f* represents the Function Number of the PF and *s* indicates which VF Slot belonging to the PF is involved. MR Functions are designated PF *h:f* and VF *h:f-s* where *h* is added to indicate which Virtual Hierarchy (VH) is involved.

This example shows a single Multi-Root Topology. TLPs inside the MR Topology are labeled with the VH they belong to. TLPs outside the MR Topology belong to a single VH and no label is needed. The MR Ingress point is the point where a TLP first encounters an MR Topology. The MR Egress point is where a TLP Exits an MR Topology. These points exist inside some MR component (in this example, the Switches and Devices X and Y).

In this example, all Root Ports use PCIe protocol. Each Root Port is the root of a PCIe Hierarchy. There are three VHs (A, B, C) each associated with one of the root Ports. Hosts A and C are running a Hypervisor that supports SR sharing. B is running an Operating System directly (no Hypervisor is involved).

In this example, there are four Devices, one of each variety.

- Device W is a Single-Root IOV Device. It is assigned to VH A. Two System Images (SI) are in use on Host A. The Hypervisor on Host A has further assigned VF 0-1 to SI 0 and VF 0-2 to SI 1.
- Device X is using both Single Root and Multi-Root sharing. PF 0 is assigned to VH B. PF 1:0 is assigned to VH B. PF 1:0 is assigned to VH C. The Hypervisor running on Host C has further assigned VF 1:0-1 to SI 4 and VF 1:0-2 to SI 5. Host C also hosts MR PCIM software which manages The MR features of the device are managed through the Base Function which is assigned to VH C.
- Device Y is using only Multi-Root sharing. PF 0:0 is assigned to VH A and PF 1:0 is assigned to VH C. In VH A, the Hypervisor has further assigned PF 0:0 to SI 2. In VH C, the Hypervisor has further assigned PF 1:0 to SI 4. The MR features of the device are managed through the Base Function which is assigned to VH C.
- Device Z is a PCIe Device. It is assigned to VH C. Virtual Machine software on C has further assigned it to SI 4.

All Switches shown in the example are Multi-Root Aware (MRA). Non-MRA Switches are also possible; however, such Switches and all components below them will be associated with a single Root Port in a single VH. Note that this non-MR sub-tree can be a mixture of SR aware and non-SR aware PCIe components.

Multi-Root Aware Components enforce separation between VHs. Software running in one VH is not allowed to affect other VHs. For example, every VH has a complete and independent address space.

Figure 2-2 shows the same example as Figure 2-1 but only shows the components visible to Host A. The MRA Switches and Devices appear to software as Single Root equivalents.

Similarly, Figure 2-3 also shows the example from Figure 2-1 but shows only components visible to Host C. Note that the link between Switch 0 and Switch 1 changes direction between these views of the topology. In Multi-Root systems, the logical upstream / downstream direction of a link is a per-VH concept and is distinct from physical link direction that was established during link bring up.

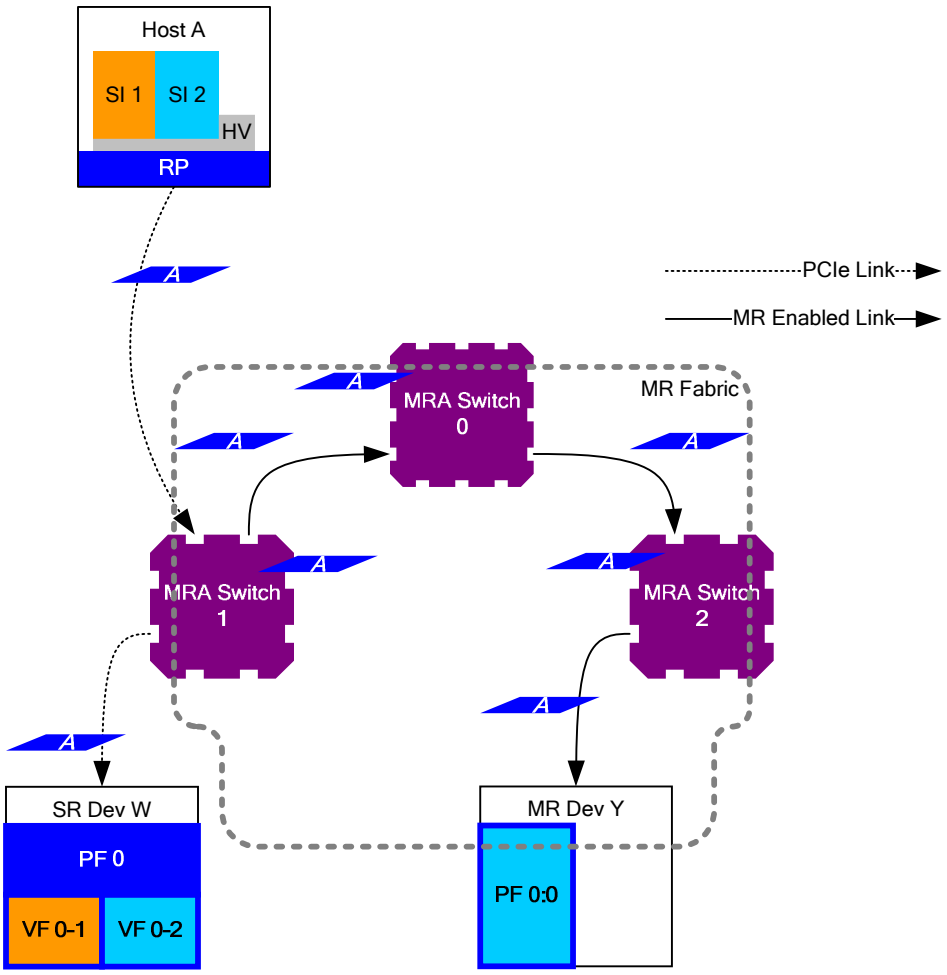


Figure 2-2: Example Multi-Root Topology as viewed from Host A

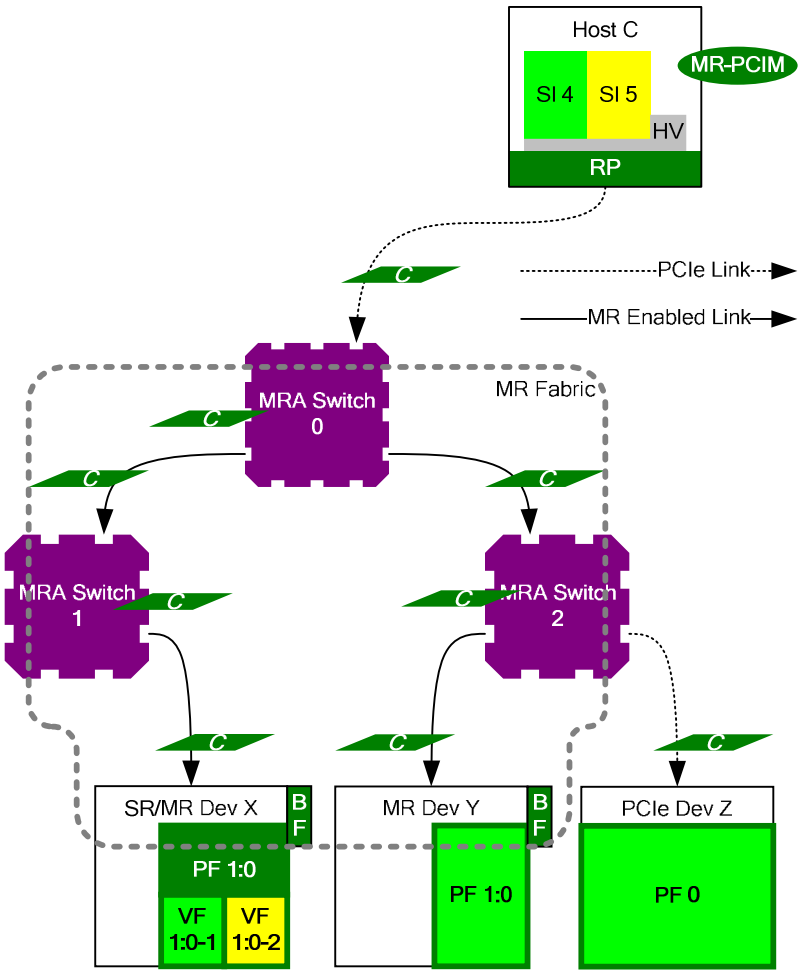


Figure 2-3: Example Multi-Root Topology as viewed from Host C

2.3.2. TLP Prefix Header

After link initialization, all links use PCIe protocol. MR-PCIM enables the MR Encapsulated Link protocol using the MR-IOV capability block located in configuration space of components on each end of the link.

TLPs on MR Enabled Links contain a TLP Prefix as shown in Figure 2-4. This prefix is located between the Sequence Number and the PCIe TLP Header.

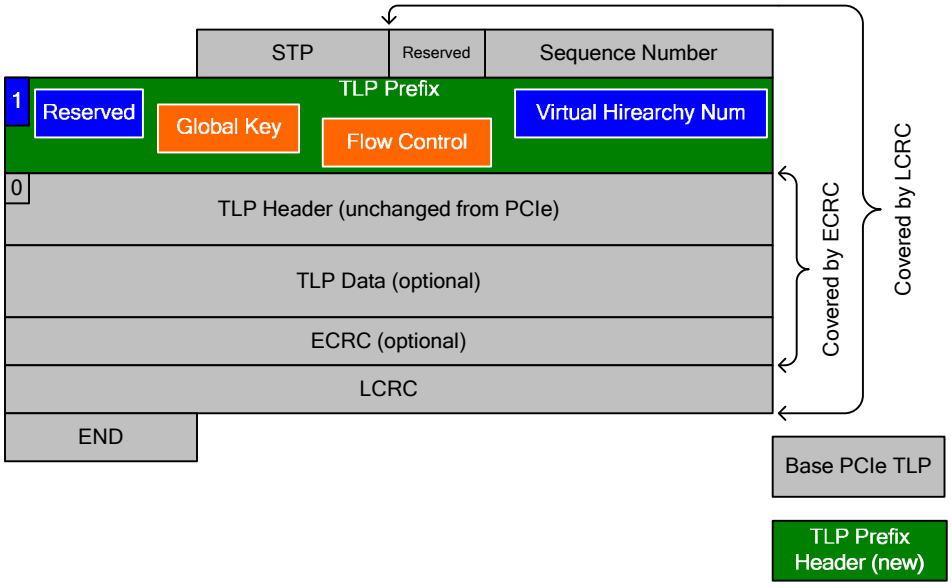


Figure 2-4: TLP Prefix Header Location

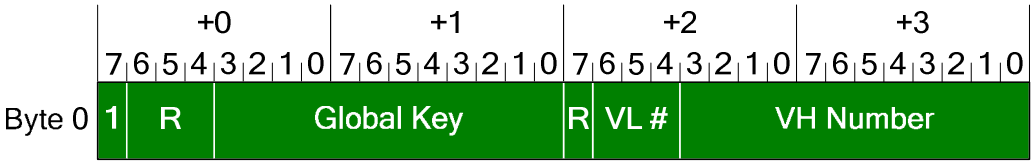


Figure 2-5: TLP Prefix Layout (tentative)

The TLP Prefix is part of the wire packet. It is covered by LCRC and is retransmitted with the rest of the TLP under the same rules. Sequence Numbers and the Ack/Nak protocol remain a per-link notion and are not affected by Multi-Root operation.

The TLP Header, Data and ECRC are not changed from PCIe TLP usage.

The layout of the TLP Prefix is shown in Figure 2-5. The first bit of the TLP Prefix is a 1 (recall that the first bit of PCIe TLPs is always a 0). Thus, the presence of the TLP Prefix can be determined by looking at the packet without needing additional link-state information. This allows MR components to process TLPs with and without TLP Prefixes when MR is being enabled. This also allows test equipment to better understand things "mid-stream".

The format of the TLP Prefix is still under discussion. Specifically, the use of the leading 1 bit vs. using a Fmt / Type value. Using a Fmt / Type value consumes 8 bits leaving only 24 for Global Key / VH / VL.

Virtual Hierarchy Numbers (VHN) are link-local. On a link that supports *N* VHs, valid VHNs are [0 .. *N*-1]. A given TLP may have different VHN values on each Multi-Root link that it traverses.

The size of the VHN field is under discussion. Figure 2-5 allows for a 12 bit value.

The Global Key field is used to guard against MR-PCIM table configuration errors and against "VH Hopping". VH Hopping is when a TLP in one VH inadvertently ends up on a different VH (due to an error inside a MR Switch). All TLPs in a VH have the same Global Key. TLPs in different VHs should have different Global Keys (protection is not provided between VHs with duplicate Global Key values). The Global Key is added to a TLP at MR Ingress based on which VH is being used (using a value configured by MR-PCIM). The Global Key value is preserved, unchanged, through MR Switches. The Global Key value is validated against the expected value at MR Egress. Any mismatch is an unrecoverable error.

The size and exact operation of the Global Key field is under discussion. Figure 2-5 allows for a 12 bit value.

The Flow Control field contains the Virtual Link Number (VL #). It contains information used to support Congestion Management and VH Isolation. See Chapter 8 Congestion Management for details.

2.3.3. MR Switch TL Processing

MR Switches implement a virtual PCIe Switch for each VH referred to as a Virtual Switch (VS). Within a Switch, every TLP is associated with a single VS. TLPs are routed inside VSs using PCIe routing (e.g. address routed, ID routed, broadcast to/from root). PCIe queuing and ordering rules apply within a VH. TLPs for unrelated VHs are unordered.

TLPs arriving on a PCIe link are assigned a VS using values programmed by MR-PCIM.

TLPs arriving on a MR Enabled Link have a VHN in the TLP Prefix once MR-PCIM has enabled MR on the sending component of the Link. This value is mapped to a VS based on values programmed by MR-PCIM (array lookup indexed by incoming VHN).

TLPs exiting a PCIe link are in a single VH. The TLP Prefix is dropped.

TLPs exiting a MR Enabled Link are tagged with a VHN assigned by MR-PCIM to the Virtual Downstream Port of the VS associated with the TLP.

2.3.4. MR TLP Dataflow Examples

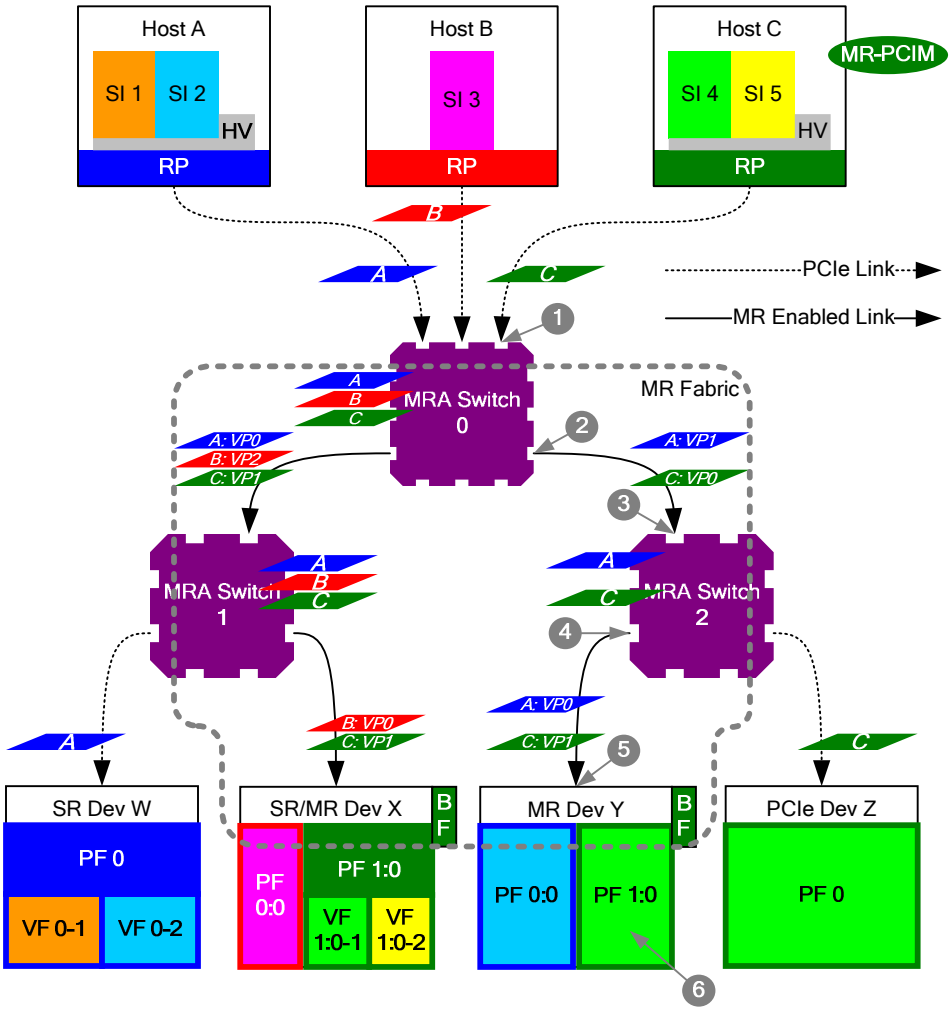


Figure 2-6: MR Dataflow Examples

Consider a Memory Read TLP initiated by SI 4 targeting PF 1:0 in EP Y:

1. The Host C to MRA Switch 0 link operated in PCIe mode. There is no TLP Prefix.
2. MRA Switch 0 has been programmed so that all TLPs arriving at Port "1" are associated with the VS for VH C.
3. The VS inside MRA Switch 0 address routes the TLP to a Virtual Downstream Port associated with physical Port "2" (the link headed towards MRA Switch 2). MR-PCIM has assigned VH 0 to this Virtual Downstream Port. The TLP exits MRA Switch 0 with a TLP Prefix having VH Number 0.

4. The TLP arrives at MRA Switch 2 Port "3" labeled as belonging to VH 0. MRA Switch 2 has been programmed so that all TLPs arriving at Port "3" labeled VH 0 are associated with the VS for VH C.

5. The VS inside MRA Switch 2 address routes the TLP to a Virtual Downstream Port associated with physical Port "4" (the link headed towards EP Y). MR-PCIM has assigned VH 1 to this Virtual Downstream Port. The TLP exits MRA Switch 2 with a TLP Prefix having VH Number 1.

6. The TLP arrives at Device Y labeled as belonging to VH 1. The Device hands the transaction to PF 1:0 for execution.

7. PF 1:0 completes the transaction and emits a completion TLP. Device Y sends this TLP out Port "5" labeled with VH 1.

8. The TLP arrives at MRA Switch 2 Port "4" labeled as belonging to VH 1. MRA Switch 2 has been programmed so that all TLPs arriving at Port "4" labeled VH 1 are associated with the VS for VH C.

9. The VS inside MRA Switch 2 ID routes the TLP to a Virtual Upstream Port associated with physical Port "3" (the link headed towards MRA Switch 0). MR-PCIM has assigned VH 0 to this Virtual Upstream Port. The TLP exits MRA Switch 2 with a TLP Prefix having VH Number 0.

10. The TLP arrives at MRA Switch 0 Port "2" labeled as belonging to VH 0. MRA Switch 0 has been programmed so that all TLPs arriving at Port "2" labeled VH 0 are associated with the VS for VH C.

11. The VS inside MRA Switch 0 ID routes the TLP to a Virtual Upstream Port associated with physical Port "1" (the link headed towards Host C). MR-PCIM has designated this link a PCIe link and has assigned this Virtual Upstream Port to it. The TLP exits Switch 0 with a no TLP Prefix.

12. SI 4 in Host C sees a completion for the Memory Read Transaction.

2.3.5. Interrupts

Interrupt processing occurs within a VH. The associated TLPs contain a TLP Prefix allowing all components to route them appropriately.

MSI and MSI-X Interrupts are indistinguishable from other Memory Write TLPs.

INTx Interrupts are represented using ASSERT_INTx / DEASSERT_INTx messages per the *PCI Express Base Specification*. In MR, these messages are queued and ordered within a VH.

2.3.5.1. INTx Device Processing

In PCIe, INTx messages are emitted by Devices. An ASSERT_INTx message is emitted when an interrupt condition is signaled. A DEASSERT_INTx message is emitted when the interrupt condition has been satisfied.

In MR, INTx messages are emitted within a VH. If a function within a VH signals or satisfies an interrupt, ASSERT_INTx / DEASSERT_INTx messages are emitted within that VH. These messages are unrelated to INTx messages issued in any other VH.

2.3.5.2. INTx Switch Processing

In PCIe, INTx messages are processed by Switches. Each downstream Switch Port tracks an internal INTx wire for each of INTA/B/C/D. These INTx wires are combined into four INTx wires at the upstream Switch Port. Transitions of the combined INTx wires trigger sending of INTx messages out the upstream Switch Port.

Similarly, In MR, INTx messages are processed by Virtual Switches. Each virtual downstream Switch Port tracks an internal INTx wire for each of INTA/B/C/D. The INTx wires are combined into four INTx wires at the virtual upstream Switch Port. Transitions of the combined INTx wires trigger sending of INTx messages out the virtual upstream Switch Port.

Switch reconfiguration will also affect INTx. For example, when a Virtual Device is unmapped from a Virtual Switch, the virtual downstream Switch Port sees DL_Down. This virtual downstream Switch Port follows PCIe rules by deasserting the internal INTx wires. If this results in a transition of the combined INTx wires a DEASSERT_INTx message is sent out the virtual upstream Switch Port.

2.3.5.3. INTx Root Port Processing

In PCIe, the Host Processor tracks the INTx assert/deassert state. If INTx is asserted, and software has not masked processing, an interrupt is sent to host processor.

In MR with a PCIe Root Port, this is unchanged. By the time the INTx message is seen by the Root Port, the TLP Prefix has been dropped and the message is indistinguishable from non-MR usage.

In MR within an MR Root Port, the INTx messages are tagged with a VH and the RP must track independent INTx assert/deassert state for each VH.

2.3.6. ARI Support

Alternate RID Interpretation (ARI) support must be provided in both virtual and physical downstream Ports of MRA Switches. Specifically, the ARI Forwarding Supported bit located in the Device Capabilities 2 register must be set and the ARI Forwarding Enable bit located in the Device Control 2 register must be implemented.

2.4. Per-VH RESET

In Multi-Root, Reset DLLPs are used to implement a per-VH equivalent of PCIe Hot Reset. DLLPs are used to ensure “quick” Reset propagation by avoiding delays introduced by TLP ordering and flow control rules.

Reset of a VH requires discarding TLPs associated with that VH. TLPs associated with VHs not in reset must not be affected. This discard process may take more time that the PCIe equivalent. For example, only some of TLPs in the retry buffer might be affected.

Reset DLLPs supports an acknowledgment protocol to mimic the TS1/TS2 ACK behavior of PCIe. This allows an upstream component to know when a downstream component has entered reset.

2.4.1. Per-VH Reset Example

Figure 2-7 shows an example MR Topology. Figure 2-8 shows Host A's view of the same topology. Table 2-1 describes the events and associated actions involved in propagating a Reset from Host A.

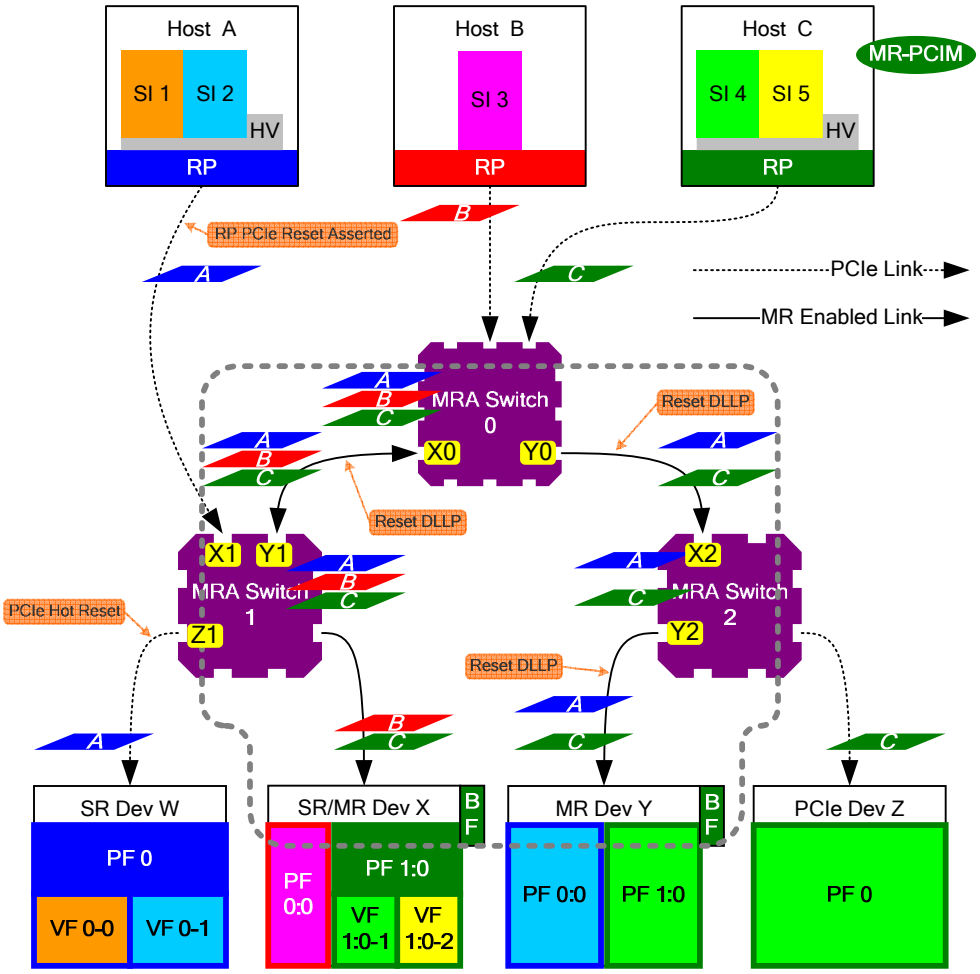


Figure 2-7: Reset DLLP Example: Topology

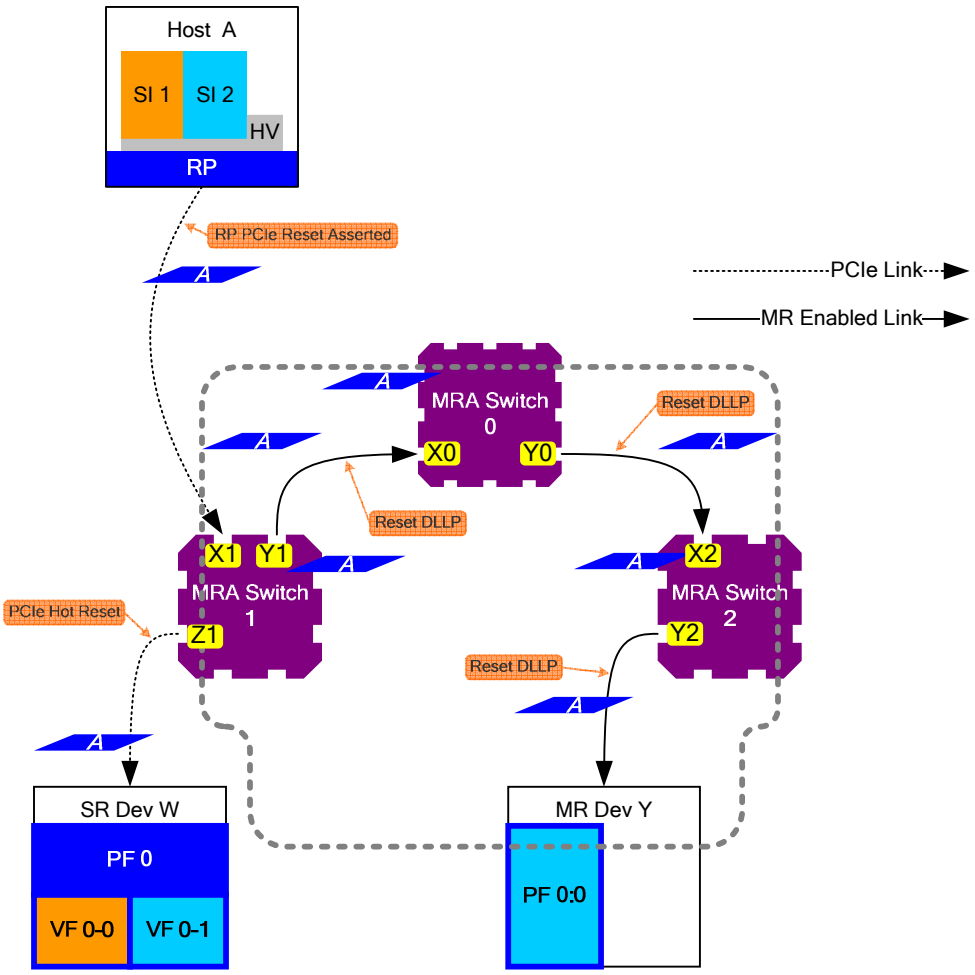


Figure 2-8: Reset DLLP Example: Host A's View

Table 2-1: Reset DLLP Example: Events and Actions

Component	Event	Action
Host A	Sends Hot Reset on RP_A to MRA Switch 1	
MRA Switch 1	Sees Hot Reset on Port X1 which is the Upstream Port of VH A	1.1 Discards all TLPs headed out Port X1 1.2 Acks Hot Reset on Port X1 (PCIe TS1/2) 1.3 Sends Hot Reset out Port Z1 to EP W 1.4 Sends Reset DLLP Request for VH A out Port Y1 to MRA Switch 0 *Unlabeled Ports not affected*
	All TLPs for VH A are discarded or marked for discard and Port X1 Retry Buffer empty	1.5 Nothing – VH A Upstream Port is not MR Link

Component	Event	Action
	Sees Reset Ack DLLP on VH A from Port Y1	1.6 Knows MRA Switch 0 will not send any more TLPs for VH A through Port Y1 1.7 Stops resending Reset DLLP for VH A out Port Y1
MRA Switch 0	Sees Reset DLLP Request for VH A on Port X0 which is the Upstream Port of VH A	0.1 Starts discarding TLPs for VH A 0.2 Sends Reset DLLP Request for VH A out Port Y0 to MRA Switch 2 *Unlabeled Ports not affected*
	All TLPs for VH A are discarded or marked for discard and Port X0 Retry Buffer contains no TLPs for VH A	0.3 Send Reset Ack DLLP out Port X0 to MRA Switch 1
	Sees Reset Ack DLLP on VH A from Port Y0	0.4 Knows MRA Switch 2 will not send any more TLPs for VH A through Port Y0 0.5 Stops resending Reset DLLP for VH A out Port Y0
MRA Switch 2	Sees Reset DLLP Request for VH A on Port Z0 which is the Upstream Port of VH A	2.1 Starts discarding TLPs for VH A 2.2 Sends Reset DLLP Request for VH A out Port Y2 to EP Y *Unlabeled Ports not affected*
	All TLPs for VH A are discarded or marked for discard and Port X2 Retry Buffer contains no TLPs for VH A	2.3 Send Reset Ack DLLP out Port X0 to MRA Switch 1
	Sees Reset Ack DLLP on VH A from Port Y2	2.4 Knows EP Y will not send any more TLPs for VH A through Port Y2 2.5 Stops resending Reset DLLP for VH A out Port Y2
EP W	Sees PCIe Hot Reset	W.1 Discards all TLPs, enters Reset W.2 Acks Hot Reset (PCIe TS1/2)
EP Y	Sees Reset DLLP Request on VH A	Y.1 Starts discarding TLPs for VH A
	All TLPs for VH A are discarded or marked for discard and no TLPs for VH A are in the Retry Buffer	Y.2 Send Reset Ack DLLP
EP X and EP Z	Nothing, not part of VH A	

2.4.2. RESET DLLP Format

Figure 2-9 shows the proposed bit encoding of the Reset DLLP.

The A bit is 0 for a Reset request (propagating downstream) and is 1 for a Reset Ack (propagating upstream).

The VH Group contains the upper bits of the VHN.

The Assert field contains one bit for each of 16 VHs within a VH Group. An Assert bit is 1 to indicate that the associated VH is in Reset, and a 0 to indicate that the associated VH is not in Reset.

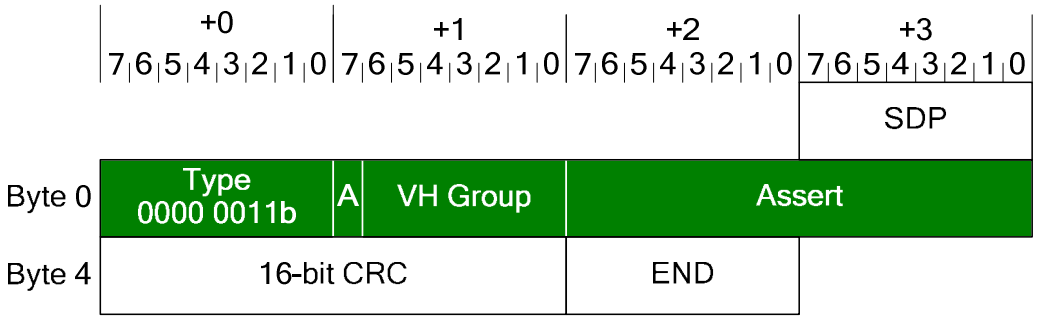


Figure 2-9: Reset DLLP

This encoding supports up to 11 bit VH numbers and would need to change if larger VH numbers are required.

Reset DLLPs can be sent at any time a link is in DL_Active. Reset Request DLLPs request a downstream link partner enter or exit Reset state on one or more VHs. Reset Ack DLLPs indicate that the downstream link partner has seen and processed the reset request.

Sending a Reset DLLP with the assert bit set is making a promise about flushing of stale TLPs. In particular, a component sending a Reset Request or Ack DLLP with Assert == 1 is claiming that all TLPs from before the VH entered Reset have been flushed (either discarded or marked for later discard). This claim includes TLPs that are sitting in the Retry buffer. Consequently, typical implementations will delay sending the Reset DLLP until the affected TLPs are retired from the Retry Buffer using normal Ack protocol (TLPs transmitted during this time will be discarded at the remote end of the link).

2.4.3. RESET DLLP Processing

The following sections describe Reset DLLP processing steps in an upstream and downstream ends of a link. Recall that upstream and downstream are relative to a VH. In particular, one end of a link may be using the upstream state machine for some VHs and the downstream state machine for other VHs.

The upstream and downstream state machines run in parallel on every VH.

2.4.3.1. Upstream State Machine

The following steps are used to enter Reset. This is triggered by a request to send a Reset DLLP on a link. This request can occur for a variety of reasons (e.g. DL_DOWN, Reset DLLP Request, or Hot Reset on an upstream Switch link, setting Secondary Bus Reset bit in some Type 1 configuration header, etc.)

1. Reset Requested.
2. Stop accepting any new TLPs to be transmitted for this VH on this link.
3. Discard any TLPs waiting to be sent for this VH on this link.
4. Wait for all TLPs that have been sent to be acknowledged and thus be retired from the Retry Buffer.

5. Send reset Request DLLP, Assert bit = 1
6. Resend Reset Request DLLP every 30 µsec for 2 ms until Reset Ack DLLP received
7. If a Reset Ack DLLP was received with Assert bit = 1, the remote end entered Reset
8. If no Reset Ack DLLP was received, retrain the link.

The following steps are used to exit Reset. This is triggered by the condition causing the entry into Reset going away (e.g. clearing Secondary Bus Reset, Physical LinkUp transitions from 0 to 1 etc.).

1. Send Reset Request DLLP, Assert bit = 0
2. Resend Reset Request every 30 µsec for 2 ms until Reset Ack DLLP received
3. If a Reset Ack DLLP has been received with Assert bit = 0, the remote end has exited Reset
4. If no Reset Ack DLLP was received, retrain the link.

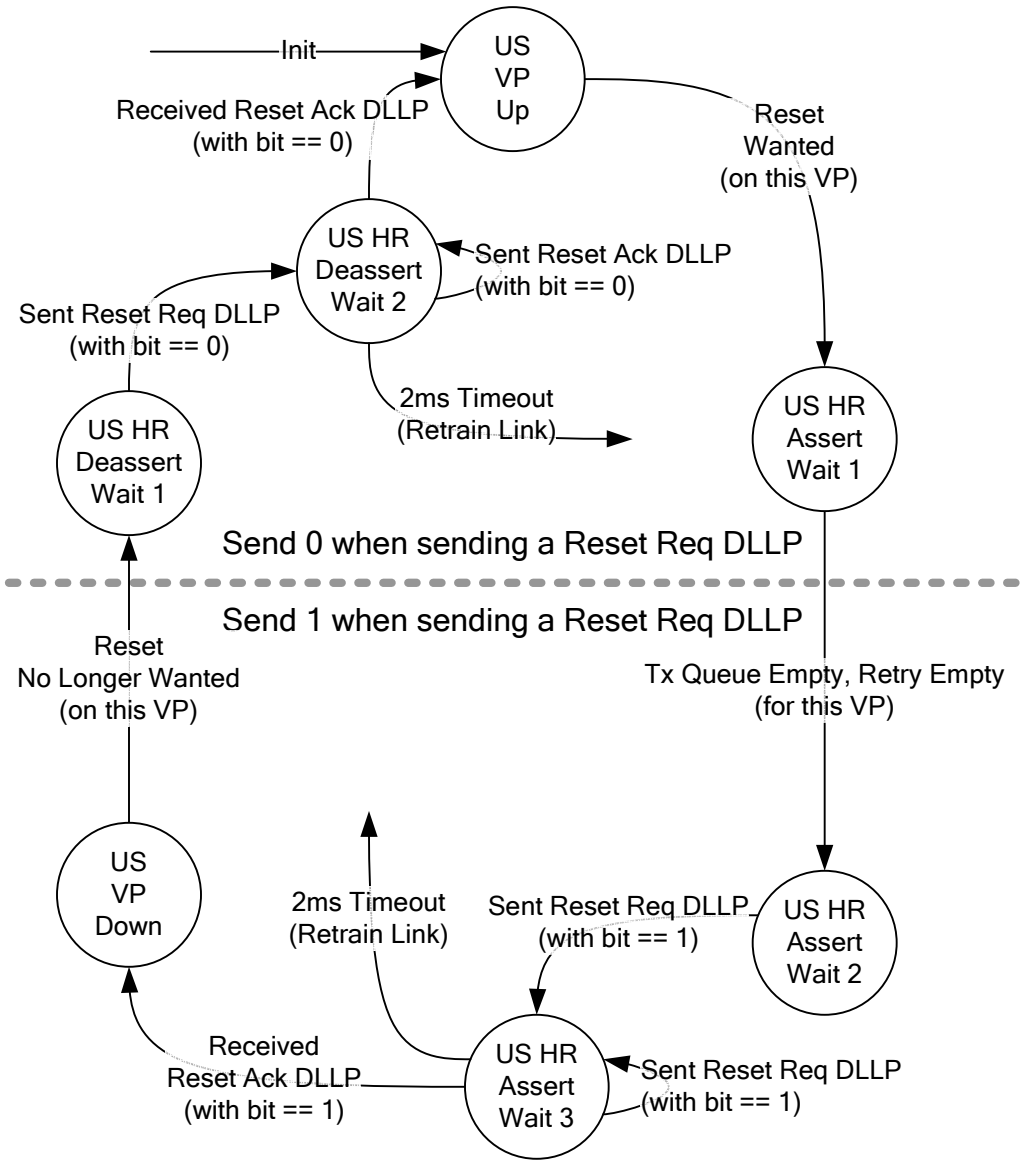


Figure 2-10: Upstream Link Partner RESET SM

2.4.3.2. Downstream State Machine

The following steps are used to enter Reset. This is triggered receiving a Reset DLLP on a link with the Assert bit = 1.

1. Reset Request DLLP received, Assert bit = 1
2. Stop accepting any new TLPs to be transmitted for this VH on this link.
3. Discard any TLPs waiting to be sent for this VH on this link.

4. Wait for all TLPs that have been sent to be acknowledged and thus be retired from the Retry Buffer.
5. Send Reset Ack DLLP, Assert bit = 1
6. Resend Reset Ack DLLP whenever Reset Request DLLP is received

The following steps are used to exit Reset. This is triggered by receiving a Reset DLLP on a link with the Assert bit = 0.

1. Reset Request DLLP received, Assert bit = 0
2. Send Reset Ack DLLP, Assert bit = 0
3. Resend Reset Ack whenever Reset Request DLLP is received

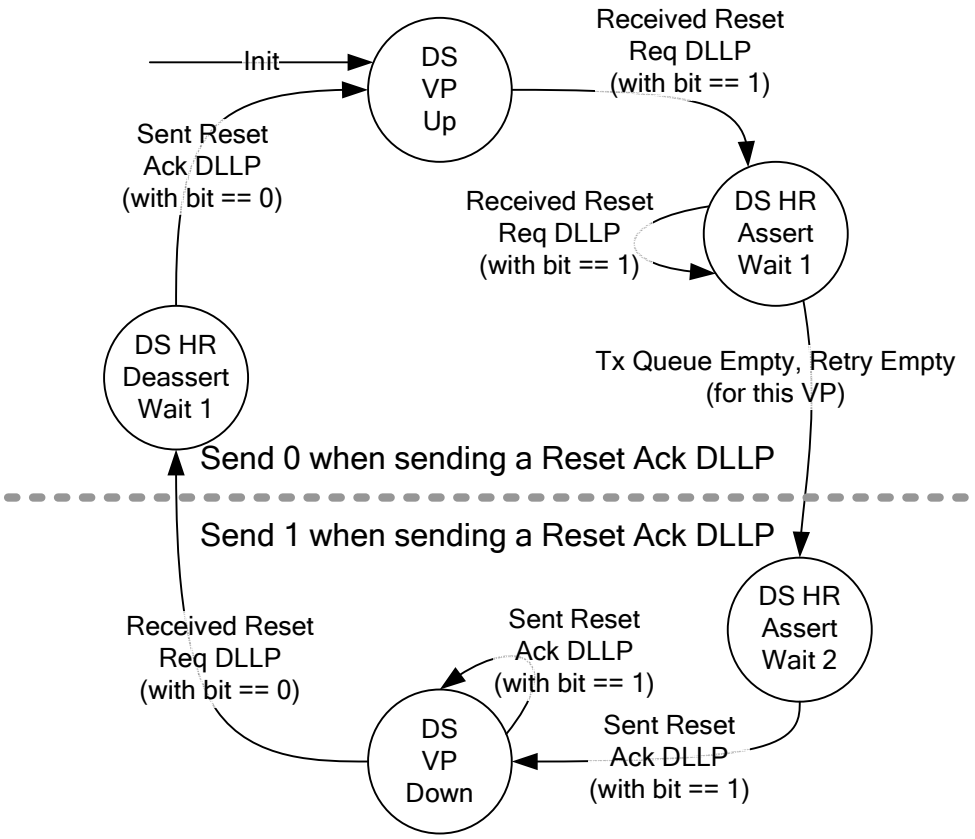


Figure 2-11: Downstream Link Partner RESET SM

2.4.3.3. Flow Control and Reset / DL_DOWN

In Multi-Root systems, Flow Control DLLPs can affect more than one VH. It is critical that a VH entering or exiting the Reset state does not disrupt other VHs. Flow Control credits on MR Enabled

Links must be returned to the originator even when the VH that originated the discarded TLPs is in Reset.

For example, suppose TLPs enter a Switch on the MR Enabled Link associated with Port 1 and the TLPs are destined for Port 2. If Port 2 is enters Reset or DL_DOWN and thus discards these TLPs, credits must be returned in the appropriate VH of Port 1. This must occur even if the associated VH at Port 1 is also in or entering reset (there may be other VHs not in reset sharing Port 1). This must occur whether or not the Link associated with Port 2 is MR Enabled.

3

3. Initialization and Resource Allocation

3.1. Requirements

- MR Devices shall operate as PCIe Devices following Reset
- MR Devices are discovered and configured using MR IOV Capabilities Configuration space
- MR Switches are discovered and configured using MR IOV Capabilities Configuration space associated with Management Ports.
- MR Switches shall have at least one Management Port.
- The specification shall allow multiple MR PCIMs to manage the system.
- The specification shall not specify how multiple MR PCIMs arbitrate for control of the MR Topology
- MR PCIM shall be able to discover and configure all MR IOV components using PCIe protocol
- MR PCIM shall optionally support mapping of VFs into VHs
- MR PCIM shall optionally support migration of VFs between VHs

3.2. MR Topology Initialization

MR Fabrics consist of a collection of the following components interconnected in a mesh topology:

- one or more MRA Switches
- one or more PCIM enabled Management Ports
- zero or more non-PCIM enabled RPs
- zero or more Devices – MRA, SR, PCIe Device[1]

Unlike conventional PCI Express, physical topology MR fabrics are not tree but a mesh which may contain loops. MR topologies have no single notion of upstream and downstream directions on

[1] Useful MR Topologies will eventually have at least one device, but this is not strictly required. In particular, no devices might be present initially with devices being added later using Hot-Add operations.

links. Logical upstream and downstream directions exist within each VH which may differ from the physical upstream and downstream directions.

This section describes how MR-PCIM configures the above collection of components to form a MR topology.

3.2.1. Fundamental Reset Initialization

MRA Switch Ports are configured into one of two Initial Port Types: *PCIM Capable Port* and *Non-PCIM Capable Port*. This configuration is set after Fundamental Reset using a Vendor Specific mechanism. The configuration mechanism can change Port type assignments but the results of such a change are not visible until the next Fundamental Reset.

The initial configuration of every MR Switch must have at least one Management Port. This port can be either a PCIe PCIM Capable Port or a Vendor Specific non-PCIe Management Port.

The initial configuration of every MR Switch may have any number of Non-PCIM Ports (including zero).

Initial Port Type is indicated by the setting of a few key characteristics of a Port. These characteristics are configurable so that MR-PCIM software can change PCIe Port behavior from the initial settings. Port Type configurability for non-PCIe Management Ports is Vendor Specific.

Implementation Note: Vendor specific settings could be used to configure an MR Switch so it operates as an SR Switch. The SR upstream Port would be a PCIM Capable Port. SR downstream Ports would be Non-PCIM Ports with their Link_Direction set as downstream and their Port VH 0 mapped to some downstream P2P Bridge of the VS associated with the upstream Port.

3.2.1.1. PCIM Capable Ports

A PCIM Capable Port i is defined as a Port with the following characteristics:

- Port[i].Link_Direction indicates an upstream Port.
- Some VS[j] has mapped {Port[i], Port VH 0} to its upstream bridge.
- The upstream P2P Bridge Configuration Header for VH 0, VS[j] has a full MR-IOV Capability.
- The bit j of the VS Authorization Bitmap is set.
- The Management VS value is Vendor Specific. It could be j or it could be some other VS.

These settings ensure that MR-PCIM *could* manage the Switch using this Port. MR-PCIM is *not required* to be present on this Port. The RP running MR-PCIM could be directly attached to this Port or could be connected indirectly using one or more MR or SR Switches.

This Port is authorized. In particular, any software claiming to be MR-PCIM will be believed. MR-PCIM can later de-authorize the Port if desired.

Note: Since a VS has a single upstream bridge, these rules imply that every Potential PCIM port will be associated with a distinct VS.

Implementation Note: PCIM Capable Ports need not be full width "expensive" Ports. They could be 1x ports intended just to support management.

3.2.1.2. Non-PCIM Capable Ports

A Non-PCIM Capable Port j is defined as a port with the following characteristics:

- Port[j].Link_Direction is Vendor Specific.
- No Authorized VS has mapped {Port[j], any Port VH} to its upstream bridge
- Mapping of Port[j] into bridges in VSs is otherwise Vendor Specific.
 - Any Port VH of Port[j] may be mapped to any downstream bridge of any VS.
 - Any Port VH of Port[j] may be mapped to the upstream bridge of any non-authorized VS. This upstream bridge may or may not have an MR-IOV capability in its Type 1 Configuration header. If it has one, only selected fields in the Type 1 header are operational and none of the MR-IOV tables located in memory are visible. See <REF> for details.
 - Port VHs of Port[j] need not be mapped into any VS.

These settings ensure that Initial MR-PCIM will never be present on this Port. This Port can be directly connected to Switches, Devices or Bridges. It can be connected, directly or indirectly via Switches RP(s) that do not run MR-PCIM after reset.

This Port is not authorized. Attempts by software attached to this Port to configure the MIOV tables of the Switch will fail. This Port can later be authorized if desired.

3.2.1.3. Non-PCIe Management Ports

A non-PCIe port may also be used to manage MR Topologies. These consist of Vendor Specific hardware that appears to a MR Switch as an upstream Port. Such Ports have a Physical Port Table entry with the Non-PCIe bit set.

This Vendor Specific hardware allows MR-PCIM to issue and respond to the subset of PCIe transactions needed to manage the MR Topology.

- Such hardware must allow MR-PCIM to issue:
 - Configuration Read and Write Requests (Type 0 and Type 1) or sizes 1, 2 or 4 bytes, naturally aligned.
 - Memory Space Read and Write Requests (32 bit and 64 bit addressing) of sizes 1, 2, 4 and 8 bytes, naturally aligned.
 - Message Transactions normally issued by Root Ports (e.g. PME messages).
- Such hardware must allow MR-PCIM to respond to:

- Completions related to the above (including completion status and support for Completion Timeout).
- Posted Memory Write transactions for MSI Interrupts
- Message Transactions (e.g. INTx, PME, Errors, …)

This is the minimum support needed to configure the MR Topology. Additional support may be required if MR-PCIM uses the MR Topology for general I/O (e.g. Logging). Additional support may also be required if Vendor Specific device management software also needs to use this port.

A Non-PCIe Management Port i is defined as a Port with the following characteristics:

- The Port is always upstream. Port[i] Direction bits are TBD.
- The Port operates in SR mode (the link can not be MR Enabled).
- Some VS[j] has mapped {Port[i], Port VH 0} to its upstream bridge. This mapping may be fixed.
- The upstream P2P Bridge Configuration Header for VH 0, VS[j] has a full MR-IOV Capability.
- The bit j of the VS Authorization Bitmap is set. This bit may be read only.
- The Management VS value is Vendor Specific. It could be j or it could be some other VS.

These settings ensure that MR-PCIM *could* manage the Switch using this Port. MR-PCIM is *not required* to be present on this Port.

This Port is authorized. In particular, any software claiming to be MR-PCIM will be believed.

Note: Since a VS has a single upstream bridge, these rules imply that every Non-PCIe Management Port will be associated with a distinct VS.

3.2.1.4. Initial Port Type Example

Figure 3-1 shows an example MR Topology. Both Switches are MRA Switches. The three green Root Ports labeled RP 0, RP 1 and RP 2 are capable of running MR-PCIM software and configuring the topology. The blue RP 3 is not permitted to run MR-PCIM. All devices are either SR or MR capable.

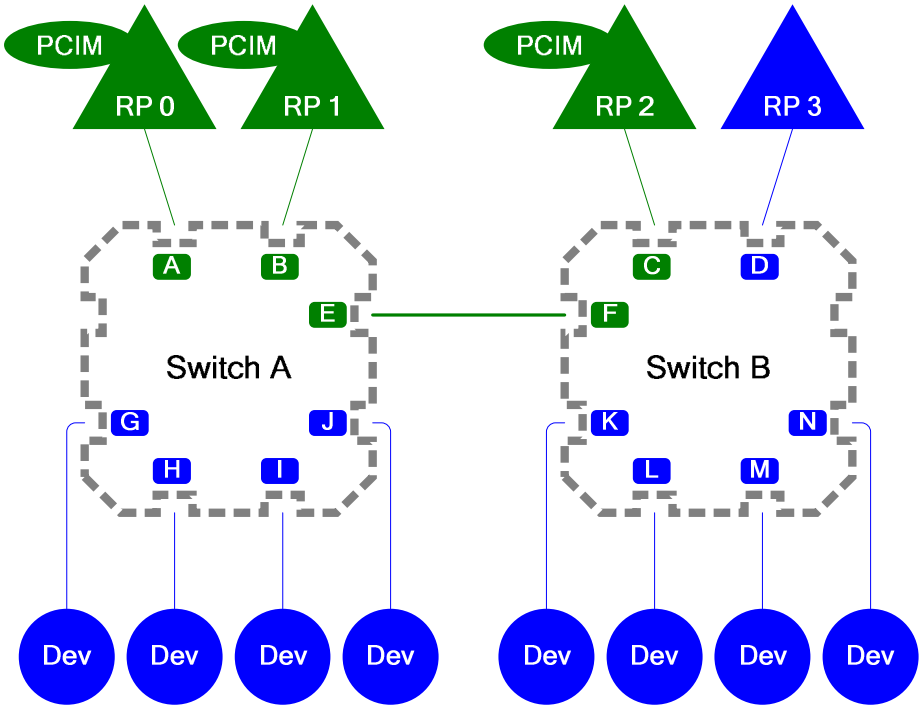


Figure 3-1: Example MR Topology

In this example, assignments are show in Table 3-1.

Table 3-1: Port Types – Example MR Topology

Port	Initial Port Type	Reason
A, B, C	PCIM Capable Port	After Reset, RP 0, RP 1 or RP 2 are allowed to run MR-PCIM
D	Non-PCIM Capable Port	After Reset, RP 3 is not allowed to run MR-PCIM
E	PCIM Capable Port	If RP 2 runs MR-PCIM, this setting allows it to manage Switch A after Reset
F	PCIM Capable Port	If either RP 0 or RP 1 runs MR-PCIM, this setting allows it to manage Switch B after Reset
G, H, I, J	Non-PCIM Capable Port	Devices never run MR-PCIM
K, L, M, N	Non-PCIM Capable Port	Devices never run MR-PCIM

3.2.2. MR-PCIM Topology Discovery

After Fundamental Reset, a Vendor Specific mechanism selects the Initial MR-PCIM location. This location can be either a RP or a non-PCIe Management Port. Only the selected Port is allowed to access the MR topology; a Vendor Specific mechanism prevents other RPs and Management Ports from viewing the MR topology until MR-PCIM has configured it.

The Initial MR-PCIM software must be able to manage every MR Switch in the MR Topology. In other words, every MR Switch must have at least one Management Port connected to the Initial PCIM location, either directly or indirectly via additional Switches.

If the selected Initial MR-PCIM is a RP that is directly connected to a PCIM Capable Port on some MR Switch, the Link_Direction of this link will allow it to train when connected to an RP. If the selected Initial MR-PCIM uses a non-PCIe Management Port, the Link_Direction is always upstream. In either case, the Link is not MR Enabled so all transactions occur in Link VH 0.

The upstream P2P Bridge mapped to VH 0 is authorized and contains the MR-IOV capability. MR-PCIM uses this capability to configure the Switch.

MR-PCIM invents a unique number for the Switch and writes it to the MR Switch Number field. This number is by used MR-PCIM to detect topology loops during the enumeration process.

MR-PCIM then uses the Physical Port table to enable additional links and to manage their Link_Direction. MR-PCIM might use Vendor Specific of knowledge of the topology to aid in this process if, for example, a system is using Link_Direction to prevent link training and thus hold off some RPs from seeing the hierarchy until it is configured.

For each Port that trains as a downstream Port, MR-PCIM will map the Port into the MR-PCIM VS (if it was not already mapped) and configure the downstream bridge to allow PCIe Configuration transactions on it.

For each downstream Port of the VS, MR-PCIM probes the configuration header of the component attached to the Port.

- If the component is a Device, Bridge or SR Switch, further enumeration is not needed at this point.
- If the component is a "new" MR Switch because the Switch's MR Switch Number field has a number not assigned by MR-PCIM in this enumeration cycle, the enumeration process repeats to configure the new Switch.
- If the component is an "old" MR Switch that MR-PCIM has seen before, the MR Switch Number and connection information is noted but further enumeration is not needed via this connection. Note that enumeration of the "old" Switch may not be complete, but it will be completed using other links into the Switch.

3.2.2.1. MR-PCIM Topology Discovery Example

Figure 3-2 expands on the earlier example shown in Figure 3-1 adding possible initial link directions and adding an additional link between the Switches.

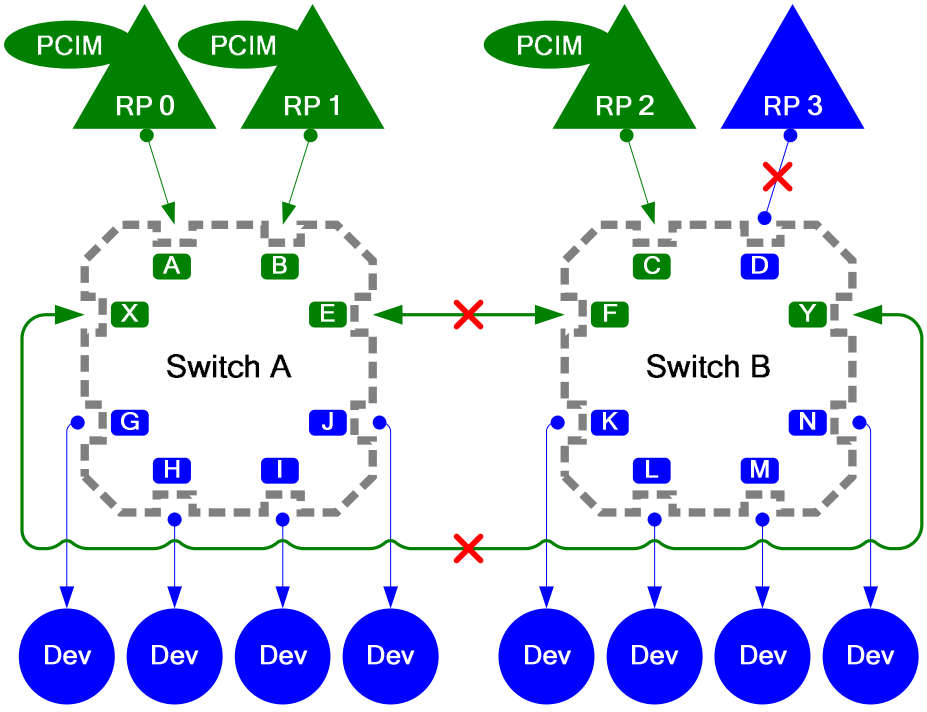


Figure 3-2: Example MR Topology with Initial Link Directions

The Links between Ports E and F and between ports X and Y do not train since each Link consists of two upstream Ports. The link between RP3 and D does not train since it consists of two downstream Ports. All other links can train after reset.

Suppose RP0 is selected as the location of the Initial MR-PCIM, Vendor Specific mechanisms are used to prevent RP 1 and RP 2 from enumerating the topology and attempting to become MR-PCIM.

MR-PCIM software operating in RP0 could proceed as follows:

1) Reads Type 1 Configuration Header at Port A, detects MR-IOV capability indicating an MR Switch. Detects that Port A, VH 0 was mapped to VSn.
2) Assigns MR Switch Number 42 to Switch A.
3) Notices link attached to Port E detected something present but did not train. Switches Link_Direction for link E to downstream thus allowing the E to F link to train.
4) Notices link attached to Port X detected something present but did not train. Switches Link_Direction for link X to downstream thus allowing the X to Y link to train.
5) Notices that Links G, H, I, J, E and X trained as downstream. If needed, maps VH 0 of these links into downstream P2P Bridges of VS n (they might already be mapped in VS n since initial mapping is Vendor Specific).
6) Enumerates the Devices at G, H, I and J. Reads the Configuration Headers and determines that these are not MR Switches so no additional enumeration is needed at this time.
7) Enumerates Switch B using Port E. Reads the Type 1 Configuration Header of Port F, detects MR-IOV capability indicating an MR Switch. Detects that Port F, VH 0 was mapped to VS m.
8) Assigns MR Switch Number 86 to Switch B.
9) Notices that links K, L, M and N trained as downstream. If needed, maps VH 0 of these links into downstream P2P Bridges of VS m.
10) Enumerates the Devices at K, L, M and N. Reads the Configuration Headers and determines that these are not MR Switches so no additional enumeration is needed at this time.
11) Notices link attached to Port D detected something present but did not train. Switches Link_Direction for link D to upstream thus allowing the D to RP 3 link to train. This step might be delayed until later if preventing link training was the Vendor Specific mechanism used hold off RP 3 from enumerating the topology.
12) Enumerates Switch B this time using Port X. Reads the Type 1 Configuration Header of Port F, detects MR-IOV capability indicating an MR Switch. The previously assigned MR Switch Number of 86 indicating that Port X is a second path to Switch B. Since Switch B has been (or is still being) enumerated using Port E, no further enumeration is needed using Port X.

This is an example; other enumeration orderings are also valid. Note that the link directions of the E to F and X to Y links depend on the chosen ordering. If, for example, Step 4 were deferred till after enumeration of Switch B, the X to Y link could train in the opposite direction.

Note: MR-PCIM assigns MR Switch registers locations in PCIe Memory Space during this discovery process. These assignments may change in subsequent software.

3.2.3. MR-PCIM Failover

There is a single *active* MR-PCIM in a topology. This MR-PCIM receives all "route to PCIM" errors and events. The VS associated with this MR-PCIM is programmed into each Switch's Management VS register.

There may be multiple Authorized VSs. Software running in these VSs is allowed to view and change the Switch MR-IOV configuration. To avoid confusion, coordination is needed between such software but is not specified by this specification.

MR-PCIM operating in any Authorized VS may become the active MR-PCIM simply by changing the Management VS register in the affected Switch(s). This feature can support failover from one MR-PCIM to a backup MR-PCIM.

Mechanisms used to detect MR-PCIM failure, to select the new MR-PCIM location, to initiate the failover, etc. are Undefined by this specification.

Messages to MR-PCIM are PCIe "Route to Root" within the Management VH (or the VH associated with the Management VS)of the component creating the message. Switches forward such messages within that VH unaffected by Management VS settings of the Switch.

Note: Ordering is not guaranteed between messages to MR-PCIM that exist in multiple VHs caused by changes to the message creating component's Management VH/VS value.

3.3. MR Device Initialization

After Conventional Reset, MR Devices operate using the PCIe protocol. At this point in time, a Device appears as one or more Physical Functions (PFs). PF 0 contains an MR-IOV Capability in the Type 0 Configuration Header.

When MR IOV Enable is set in the MR-IOV Capability of PF 0, the initial PFs become attached to VH 0 and additional PFs are visible, one set for each VH. The MR-IOV Capability block attached to PF 0:0 is now known as the Base Function (BF). This Base Function is also accessible in all Authorized VHs.

Attached to each PF is an optional collection of Virtual Functions. These are described in the SR-IOV Specification.

PFs, BFs and VFs are designated as follows:

> **BF** indicates the Base Function. It does not have a number since each Device only has one BF.
>
> **PF *f*** indicates a PF at Function number *f* (*f* must be between 0 and 255). This nomenclature is used for SR systems or for the SR view of a single MR VH.
>
> **PF *h:f*** indicates a PF within an MR system at Function number *f* in VH *h* (*h* must be between 0 and the maximum VH number in use on the link).
>
> **VF *f-s*** indicates VF number *s* attached to PF number *f* (*s* must be between 1 and the number of VFs in use for PF *f*). This nomenclature is used for SR systems or for the SR view inside a single VH.

VF *h:f-s* indicates VF number *s* attached to PF number *f* in VH *h*.

In addition, the optional VF Mapping and VF Migration features use the terms Logical Virtual Function (LVF) and Mission Virtual Function (MVF). These are similarly designated as follows:

LVF *f-s* indicates LVF table slot number *s* attached to PF number *f*.

MVF *f-s* indicates MVF number *s* attached to PF number *f*. MVFs do not have a VH (a VH is associated the LVF that an MVF is mapped to, see below).

During topology enumeration, MR-PCIM detects MR Devices by noticing the presence of an MR-IOV Capability in PF 0's Configuration Header. Using this MR-IOV Capability, MR-PCIM can cause a Device to operate in MR Mode.[2]

Initializing and managing a Device in MR mode involves managing four aspects of the Device.

- Enabling MR operation: This involves configuring the number of VHs used and turning on MR-IOV mode. It also involves specifying which VH(s) are allowed to manage the Device and where messages to MR-PCIM should be sent.
- Enabling and managing the optional MR flow control features: This involves configuring the number of Virtual Links used, configuring the number of VCs offered in each VH, configuring the {VH, VC} to VL mapping hardware, configuring any VH to VL arbitration hardware and configuring any VL to Link arbitration hardware.
- Enabling and managing the optional VF Mapping features: This involves configuring the number of LVFs offered by each PF in each VH and configuring each PF's LVF to MVF map.
- Enabling and managing the optional VF Migration features: This involves leaving "holes" in the LVF to MVH map to support migration, enabling VF Migration, responding to requests for initiate VF migration and interacting with SR-PCIM software to accomplish VF migration.

These aspects will be described separately in the following sections.

3.3.1. Enabling MR Operation

Initially MRA Devices operate in SR mode. They have one or more of Physical Functions (PFs) and operate as a PCI Express multi-function device.

In MRA Devices, Function 0 contains an MR-IOV Capability block. There are a few key registers in this Capability used in enabling MR Operation.

- The MR-IOV Capabilities register indicates which optional features are implemented by the Device.
- The MaxVH register indicates the number of VHs supported by Device hardware. The value is Vendor Specific and must not change except after Fundamental Reset of the Device.

[2] MR-PCIM could opt to leave a Device in SR mode and ignore the Device's MR features. If this case, the usual SR configuration is sufficient and no additional configuration is required.

- The NumVH register indicates to the Device how many VHs are going to be used by MR-PCIM. Software should set this value based on the value of MaxVH, on the number of VHs implemented at the upstream end of the link and on the number of VHs needed by the system. The NumVH value must not change while MR-IOV Enable is set.
- The MR-IOV Enable bit allows the Link to use the MR protocol. There is an equivalent bit at each end of a Link. Since it is not possible to simultaneously set enable bits at both ends of a link, MR components must always accept a subset of MR protocol even when this bit is clear. Software must not clear this bit once it has been set (the bit can only be cleared by a Conventional Reset of the Device).
- The MR-IOV Enable Busy bit indicates that the Device is responding to software setting the MR-IOV Enable bit. This bit allows a Device time to reconfigure itself during the SR → MR transition. There is a similar bit in the SR specification and the exact meaning of the SR bit is still under discussion. Semantics of this bit will track the decisions made for SR.
- The Management VH field indicates the VH that should receive all messages that are sent to MR-PCIM. This VH is implicitly Authorized to manage the Device as well.
- The Authorized VH Bitmap indicates which VHs are allowed to manage the Device. Writes of MR-IOV data structures from non-Authorized VHs are ignored. Reads of MR-IOV data structures from non-Authorized VHs return undefined data.
- The VH Authorized bit in the MR-IOV Capabilities register indicates whether the VH reading the register is Authorized to manage the Device.
- The VH Number field indicates the VH of the reader. This register allows MR-PCIM software to determine the appropriate index to use in other tables.

3.3.2. Managing Flow Control

There are a number of fields used to manage flow control and VC to VL mapping. The details in this are still being worked out. This section describes the current thinking.

MR Flow Control uses the same concepts as PCIe flow control. TLPs can only be sent if the transmitter has sufficient available flow control credits of the appropriate flavor. The differences from PCIe include:

In MR, VCs are replaced by Virtual Links (VLs). All Flow Control information is related to VLs not VCs. VCs continue to have meaning within a VH and serve as the mechanism used to present SR software with the collection of flow control channels and the mechanism that allows SR to designate which TLPs should get use each flow control channel. As today, SR software controls the TC to VC map. MR-PCIM controls the subsequent map from {VH, VC} to VL. Performance characteristics of a VL are assigned by MR-PCIM.

If PF *h:f* supports more than one VC, the either optional VC Capability or optional MFVC Capability exists in the PF *h:f* Type 0 Configuration Header. The values presented are managed by MR-PCIM as follows:

- The PF *h:f* Extended VC Count value is configured by MR-PCIM. This value indicates to SR software how many VC Resources are usable.
- The PF *h:f* Low Priority Extended VC Count value is configured by MR-PCIM. This value indicates to SR software which VC IDs should be viewed as higher priority.
- The PF *h:f* VC to VL map is configured by MR-PCIM. This map converts VC IDs chosen by R-software into the corresponding VL numbers. This is a full eight entry map since all VC ID values are legal in current PCIe.
- The PF *h:f* VC Enabled fields allows MR-PCIM to determine which VC Resources are enabled by SR software.
- The PF *h:f* VC ID fields allows MR-PCIM to determine the VC ID assignments made by SR software.
- The PF *h:f* TC to VC map field allows MR-PCIM to determine the mapping between TCs and VCs made by SR software.

3.3.3. Managing VF Mapping

SR-PCIM software can optionally see a collection of Virtual Functions attached to each PF. In MR, these VFs are known as Logical Virtual Functions (LVFs). The Device implements a number of underlying Mission Virtual Functions (MVFs). MVFs are mapped into LVFs. This mapping can be fixed or variable.

Fixed Mapping is Vendor Specific. A Device assigns MVFs to PFs in a Vendor Specific manner. The assignment may not change except following Fundamental Reset. VF Mapping and VF Migration are not supported and the associated registers are Undefined.

Variable Mapping uses a set of mapping tables controlled by MR-PCIM. These tables allow MR-PCIM to (1) control the number of LVFs assigned to each PF, (2) specify which MVFs (if any) are assigned to each LVF and (3) detect how many VFs SR-PCIM has said it will use.

Figure 3-3 shows an example Variable Mapping setup. There are 5 VHs, each with a single PF at Function 0.

- VH 0 has been assigned 8 LVFs and 4 MVFs. LVF 0-0 is associated with PF 0:0, LVF 0-1 thru LVF 0-7 are associated with the 7 VFs. There are 4 VF holes (VF 0:0.4 thru VF 0:0.7 a.k.a. LVF 0-4 thru LVF 0-7) meaning that, if SR-PCIM enables VF Migration, up to 4 MVFs can be migrated in to VH 0.
- VH 1 has been assigned 4 LVFs and 4 MVFs. All LVFs are populated meaning that no migration in is possible unless an MVF first migrates out of the PF and/or VH.
- VH 2 has been assigned 7 LVFs and 5 MVFs. Like VH 0, holes were left for possible VF Migration in to the VH.
- VH 3 has been assigned 3 LVFs and 3 MVFs, Like VH 1, no migration in is possible unless an MVF first migrates out of the PF and/or VH.

- VH 4 has been assigned 1 LVF and 1 MVF. No VFs are assigned and no migration is supported unless the MVF first migrates out of the PF.

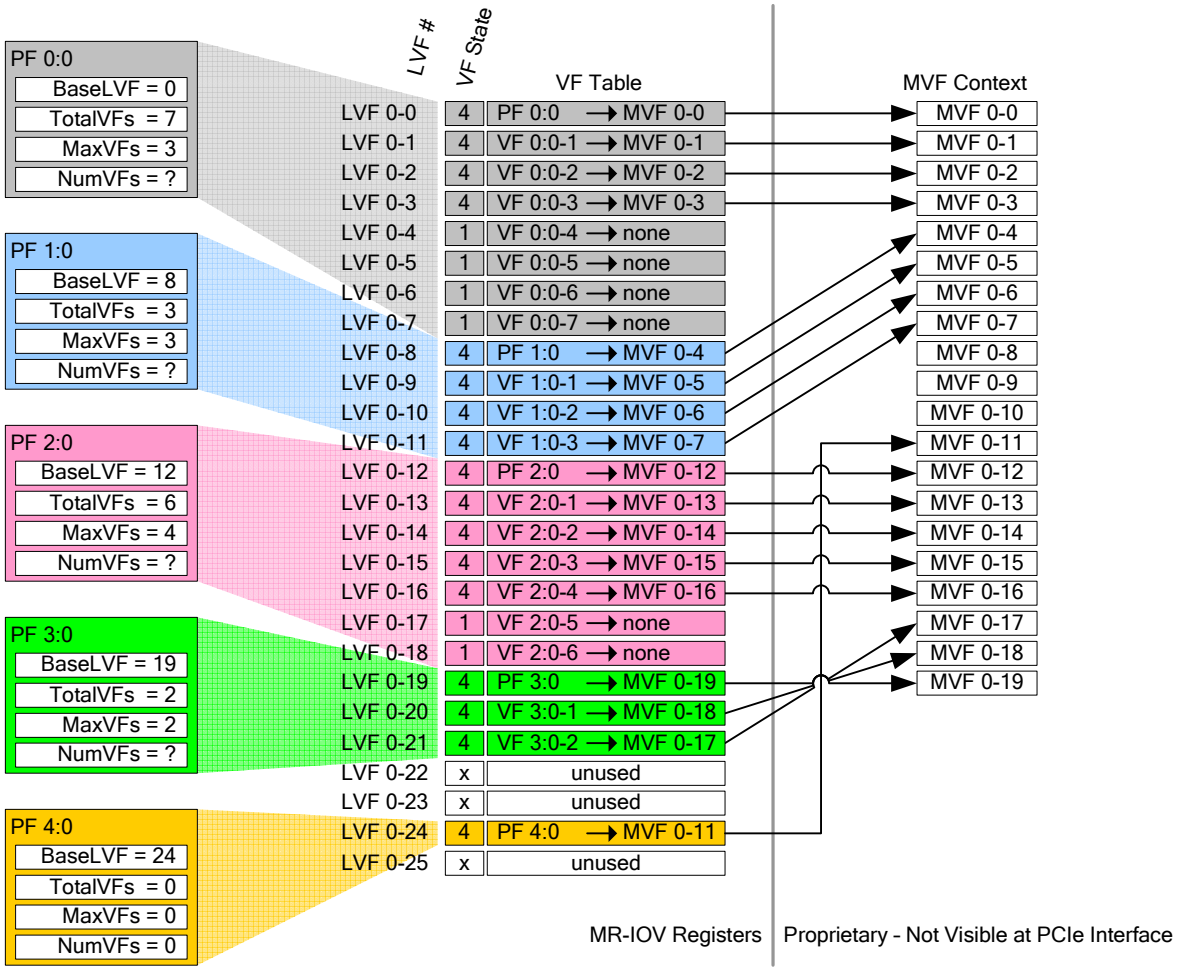


Figure 3-3: Example Mapping of VFs associated with PF 0

The VF State table is used to coordinate migration of an MVF in or out of a VH. If migration is not supported, the VF State table is not used. The VF State values shows in Figure 3-3 are the required initial values used if Migration is supported See Section 3.3.4 Managing VF Migration for details.

MR-PCIM manages the LVF slot allocation to PFs using the Base LVF and Total VFs registers. Base LVF indicates the first LVF slot associated with the PF. Base LVF + Total LVFs indicates the last LVF slot associated with the PF. The LVF slot designated by Base LVF contains the MVF associated with the PF. The LVF slot designated by Base LVF+1 contains the MVF associated with VF *x*.1 of PF *x*, etc. Every PF has at least one LVF slot.

The number of populated LVFs offered to SR-PCIM is contained in MaxVFs. If SR-PCIM does not enable VF Migration, only these slots are used and any additional unpopulated slots remain used.

Initially, MR-PCIM must populate LVFs for a given PF using the lower numbered VFs first. Holes left for migration (if any) follow the last populated LVF.

Before it enables SR-IOV operation, SR-PCIM in each VH must set the NumVFs value to indicate the number of VFs it wishes to use. The value set must be less than or equal to TotalVFs. If VF Migration not enabled by SR-PCIM, the value set must also be less than or equal to MaxVFs.

A mechanism will be provided to support Function Assignment Grouping. This addresses the issue of a collection of "related" functions that must be mapped together so that a given PF sees all functions in the assignment group at consistent VF numbers or the PF sees none of the assignment group. An example of a device that can use this feature is the de-facto standard controllers for USB 1.0 and USB 2.0. These controllers offer different programming models using different PCI functions but refer to the same underlying hardware. This mechanism is also needed in SR and the MR specification will borrow from the mechanism being defined for SR.

3.3.4. Managing VF Migration

In Multi-Root systems, VFs can be migrated between VHs. VF Migration does not occur in Single-Root only systems.

SR and MR support for VF Migration is *Optional*. VF Migration for PF *f* is only possible if all three of the following are true:

- The PF *f* provides VF Migration hardware (i.e. the VF Migration hardware Present bit is set).
- MR-PCIM software controlling PF *f* has set the VF Migration Capable bit.
- In at least one VH (*h*), SR-PCIM software controlling PF *h:f* has set the VF Migration Enabled bit.

VF Migration centers around two data structures:

- The VF Table maps LVFs onto MVFs. When a VF State is inactive, this table can be changed to implement the change.
- The VF State array manages the SR and MR combined view of the state of a VF. This table can be used to signal graceful remove and adds of VFs to / from VHs.

VF migration follows the state diagram shown in Figure 3-4. The state values shown are contained in the VF State array entry associated with the VF. State transitions indicated by solid lines are initiated by MR software by writing the new state value to the VF State array. State transitions indicated by dashed lines are initiated by SR software using mechanisms described in the SR Specification.

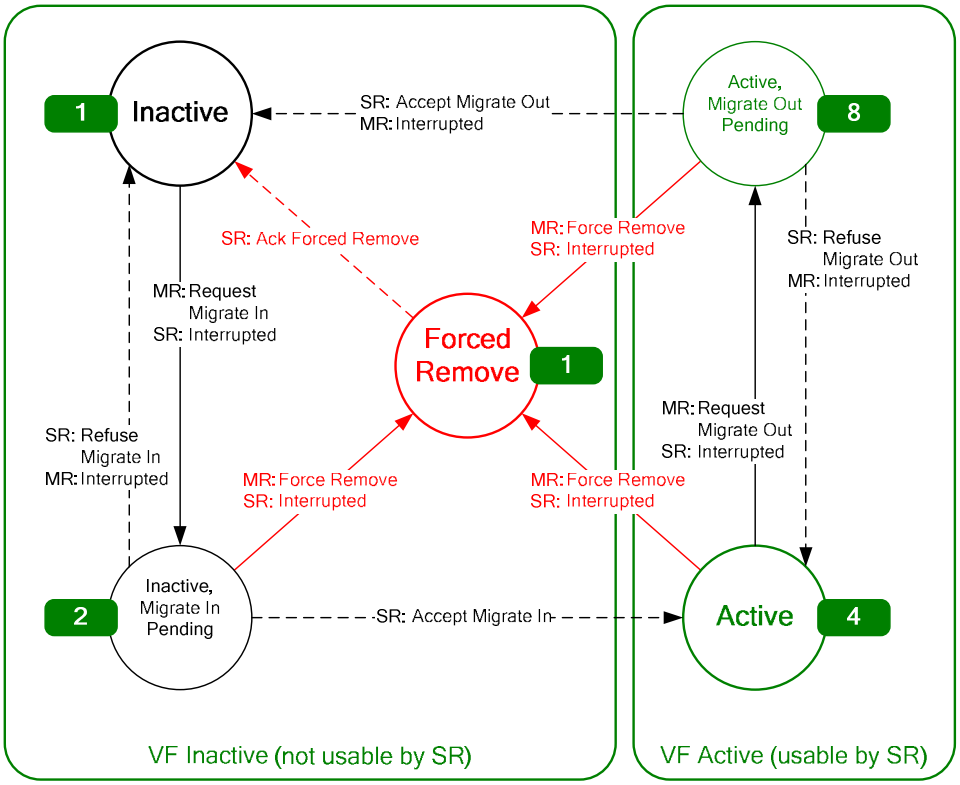


Figure 3-4: VF Migration State Diagram

The following state transitions may be initiated by MR software by writing the appropriate VF State entry. Any other writes are ignored and no state transition occurs. Software should write 0 when no state transition is intended.

Table 3-2: Valid MR State Transitions for VF Migration

Current State	Written State	Meaning
1	2	Initiate Migrate In
4	8	Initiate Migrate Out
2	1	Initiate Forced Remove
4	1	Initiate Forced Remove
8	1	Initiate Forced Remove

VFs that are in an inactive state are not usable by software in any way. Configuration, Trusted Configuration, IO and Memory Transactions targeting the associated RID return UR. Within nnn milliseconds of transitioning to an Invalid state, a Device must guarantee that it will not issue TLPs using the indicated RID.

State transitions with the notation "MR: Interrupted" cause the VF Migration Interrupt Status bit to be set. If VF Migration Interrupt Enable is set, this in turn causes an MSI to be queued to MR-PCIM. MR-PCIM software can then scan the VF State array to comparing the current state or each entry to a previously saved state. For Devices with a large number of VFs, this scan can be a

performance issue – a future version of this specification may define mechanisms to reduce the performance impact of such a scan.

State transitions with the notation "SR: Interrupted" will cause a similar interrupt status bit to be set in the SR universe.

The following steps are use by MR-PCIM to migrate a VF from one VH to another.

1. Request arrives from higher level software requesting that **VF *h:f-s*** be migrated to **VF *a:f-c***.
2. Initiate a *Migrate Out* operation in VH *h* by writing VF State entry associated with VF *h:f-s* to the value 8.
3. Wait for SR-PCIM to stop using the VF and to indicate so by transitioning the VF State to 1. This transition can raise an interrupt to MR-PCIM.
4. Save the value of the VF Table Entry associated with VF *h:f-s*.
5. Change the VF Table Entry associated with VF *h:f-s* to none to indicate and empty slot
6. Ensure that the VF Table entry associated with VF *a:f-c* is empty.
7. Change the VF Table Entry associated with VF *a:f-c* the value saved in step 3
8. Initiate a *Migrate In* operation in VH *a* by writing the VF State entry associated with VF *a:f-c* to the value 2.
9. Wait for SR-PCIM to start using the VF and to indicate so by transitioning the VF State to 4. This transition can raise an interrupt to MR-PCIM.

In addition graceful migration described above, MR-PCIM can initiate *Forced Remove* operations on VFs. These are analogous to Surprise Remove operations in conventional PCIe. Forced Remove is initiated by writing VF State to 1. SR software is interrupted and can then emulate a Surprise Remove to the appropriate SI.

4

4. Configuration

The following sections list the configuration requirements for Base Function (BF), Physical Functions, MRA Switches and MR-PCIMs.

4.1. Requirements

- MR IOV Capability shall exist in the Configuration space for the upstream PCI-to-PCI Bridge of every VS associated with a Management Port.
- MR IOV Capability shall exist in every BF of a device.
- Additional MR IOV configuration and control structures shall exist in memory space of devices

4.2. Configuration Field Summary

The following fields are used to manage MR IOV features of a Device or Switch.

Table 4-1: MR-IOV Fields

Field Name	Width	Type	Opt	Device Usage	Switch Usage
MR-IOV Enable	1	RW	Req	BF / Control	Physical Port Table / Control
Max VH	12	RO	Req	BF / Capability	Physical Port Table / Capability
Num VH	12	RW	Req	BF / Control	Physical Port Table / Control
Link Direction Status	2	RO	Opt	n/a	Physical Port Table / Status
Link Partner Detected	1	RO	Opt	n/a	Physical Port Table / Status
Link Direction Control	2	RW	Opt	n/a	Physical Port Table / Control
Port Present	1	RO	Req	n/a	Physical Port Table / Capability
Port Enable	1	RW	Req	n/a	Physical Port Table / Control
VF Migration					
VF Migration Hardware Present	1	RO	Req	Per PF / Capability	n/a
VF Map Hardware Present	1	RO	Req	Per PF / Capability	n/a
VF Enabled	1	RO	Req	Per PF Per VH / Status	n/a
VF Migration Capable	1	RW	Opt	Per PF Per VH / Control	n/a
VF Migration Enabled	1	RO	Opt	Per PF Per VH / Status	n/a
Max LVF #	16	RO	Opt	Per PF / Capability	n/a
Max MVF #	16	RO	Opt	Per PF / Capability	n/a
Low LVF #	16	RW	Opt	Per PF Per VH / Control	n/a
Max VFs	16	RW	Opt	Per PF Per VH / Control	n/a
Total VFs	16	RW	Opt	Per PF Per VH / Control	n/a
Num VFs	16	RO	Opt	Per PF Per VH / Status	n/a
Congestion Management					
VL Arbitration Table Offset	8	RO	Opt	BF / Capability	Physical Port Table / Capability
VL Arbitration Entry Size	2	RO	Opt	BF / Capability	Physical Port Table / Capability

Field Name	Width	Type	Opt	Device Usage	Switch Usage
VL Arbitration Reference Clock	2	RO	Opt	BF / Capability	Physical Port Table / Capability
VL Arbitration Capability	8	RO	Opt	BF / Capability	Physical Port Table / Capability
Load VL Arbitration Table	1	RW	Opt	BF / Control	Physical Port Table / Control
VL Arbitration Select	3	RW	Opt	BF / Control	Physical Port Table / Control
VL Arbitration Status	1	RO	Opt	BF / Status	Physical Port Table / Status
Low Priority VL Threshold	3	RW?	Opt	BF / Control	Physical Port Table / Control
Num VL	3	RO	Req	BF / Capability	Physical Port Table / Capability
Max VL	3	RW	Opt	BF / Control	Physical Port Table / Control
Extended VC Count	3	RW	Opt	Per PF Per VH / Control	VS Bridge Table / Control
Low Priority Extended VC Count	3	RW	Opt	Per PF Per VH / Control	VS Bridge Table / Control
VC Enabled	8	RO	Opt	Per PF Per VH / Status	VS Bridge Table / Status
VC ID	24 (3*8)	RO	Opt	Per PF Per VH / Status	VS Bridge Table / Status
TC to VC Map	64 (8*8)	RO	Opt	Per PF Per VH / Status	VS Bridge Table / Status
Num VC Resources Present	3	RO	Opt	Per PF Per VH / Capability	VS Bridge Table / Status
VC to VL Map	24 (3*8)	RW	Opt	BF / Control	VS Bridge Table / Control
Global Key Management					
Global Key Table Offset	32	RO	Req	BF / Capability	n/a
Global Key Check Enable	1	RW	Req	BF / Control	n/a
Global Key Table	12 * MaxVH	RW	Req	BF / Control	n/a
VS Global Key Value	12	RW	Req	n/a	VS Table / Control
VS Global Key Check Enable	3	RW	Req	n/a	VS Table / Control
MR-PCIM Authorization					
MR-PCIM VH	~ 12	RW	Req	BF / Control	n/a
Authorized VH Bitmap Offset	8	RO	Req	BF / Control	n/a
Authorized VH Bitmap	MaxVH+1	RW	Req	BF / Control	n/a
MR-PCIM VS	~ 12	RW	Req	n/a	MR-IOV Cap / Control
Authorized VS Bitmap Offset	8	RO	Req	n/a	MR-IOV Cap / Control

Field Name	Width	Type	Opt	Device Usage	Switch Usage
Authorized VS Bitmap	NumVS	RW	Req	n/a	MR-IOV Cap / Control
Switch Mapping					
Bridge Present	1	RO	Req	n/a	VS Bridge Table / Capability
VS Bridge Enabled	1	RW	Req	n/a	VS Bridge Table / Control
Bridge Port	8	RW	Req	n/a	VS Bridge Table / Control
Bridge Port VH	12	RW	Req	n/a	VS Bridge Table / Control
VS Present	1	RO	Req	n/a	VS Table / Capability
VS Enable	1	RW	Req	n/a	VS Table / Control
VS Interrupt Vector Num	11	RO	Req	n/a	VS Table / Capability
Hot Plug Signals Interface					
Hot Plug Hardware Present	1	RO	Req	n/a	VS Bridge Table / Capability
Slot Implemented	1	RW	Opt	n/a	VS Bridge Table / Control
Physical Slot Number	8	RW	Opt	n/a	VS Bridge Table / Control
Slot Power Limit Scale	2	RO	Opt	n/a	VS Bridge Table / Capability
Slot Power Limit Value	8	RO	Opt	n/a	VS Bridge Table / Capability
Hot Plug Capable	1	RW	Opt	n/a	VS Bridge Table / Control
Hot Plug Surprise	1	RW	Opt	n/a	VS Bridge Table / Control
Data Link State	1	RW	Opt	n/a	VS Bridge Table / Control
Attention Indicator State	2	RO	Opt	n/a	VS Bridge Table / Status
Attention Indicator Change	1	RW1C	Opt	n/a	VS Bridge Table / Status
Attention Indicator Change Interrupt Enable	1	RW	Opt	n/a	VS Bridge Table / Control
Press Attention Button	1	RW	Opt	n/a	VS Bridge Table / Control
Power Indicator State	2	RO	Opt	n/a	VS Bridge Table / Status
Power Indicator Changed	1	RW1C	Opt	n/a	VS Bridge Table / Status
Power Indicator Changed Interrupt Enable	1	RW	Opt	n/a	VS Bridge Table / Control
Power Controller State	1	RO	Opt	n/a	VS Bridge Table / Status

Field Name	Width	Type	Opt	Device Usage	Switch Usage
Power Controller Changed	1	RW1C	Opt	n/a	VS Bridge Table / Status
Power Controller Changed Interrupt Enable	1	RW	Opt	n/a	VS Bridge Table / Control
Power Controller Present	1	RW	Opt	n/a	VS Bridge Table / Control
Signal Power Fault	1	RW	Opt	n/a	VS Bridge Table / Control
Presence Detect State	1	RW	Opt	n/a	VS Bridge Table / Control

4.3. Device Configuration Space

In MR IOV topologies MR-PCIM must use the Base Function to discover VHs and VFs. This specification provides EP vendors with two mechanisms for configuring the VF associated with VHs: EP based VF to PF mapping and MR-PCIM based VF to PF mapping.

EP based VF to PF mapping is intended to be a simple approach that allows EP vendors to distribute VF across VH using a proprietary mechanism. Under this mechanism MR-PCIM uses the Base Function to discover the number of VH supported by the EP. MR-PCIM then uses the Base Function's Configuration Space to set the number of VHs associated with the EP allowing the EP to configure its VFs across these VHs.

MR-PCIM based VF to PF mapping is intended to be a flexible approach that, among other things, allows EP vendors to support the migration of VFs from one VH to another VH. Under this mechanism MR-PCIM uses the Base Function to discover the number of VH supported by the EP. MR-PCIM then uses the Base Function's Configuration Space to set the number of VHs supported by the EP and the Base Function's Memory Mapped Configuration Space to further configure its VFs.

The Base Function is used by MR-PCIM to discover and configure the EPs MR-IOV Capabilities regardless of the number of Physical Function Types supported by the EP. This section will describe the two types of Configuration Space and the fields used by MR-PCIM to perform Virtual Function discovery and configuration.

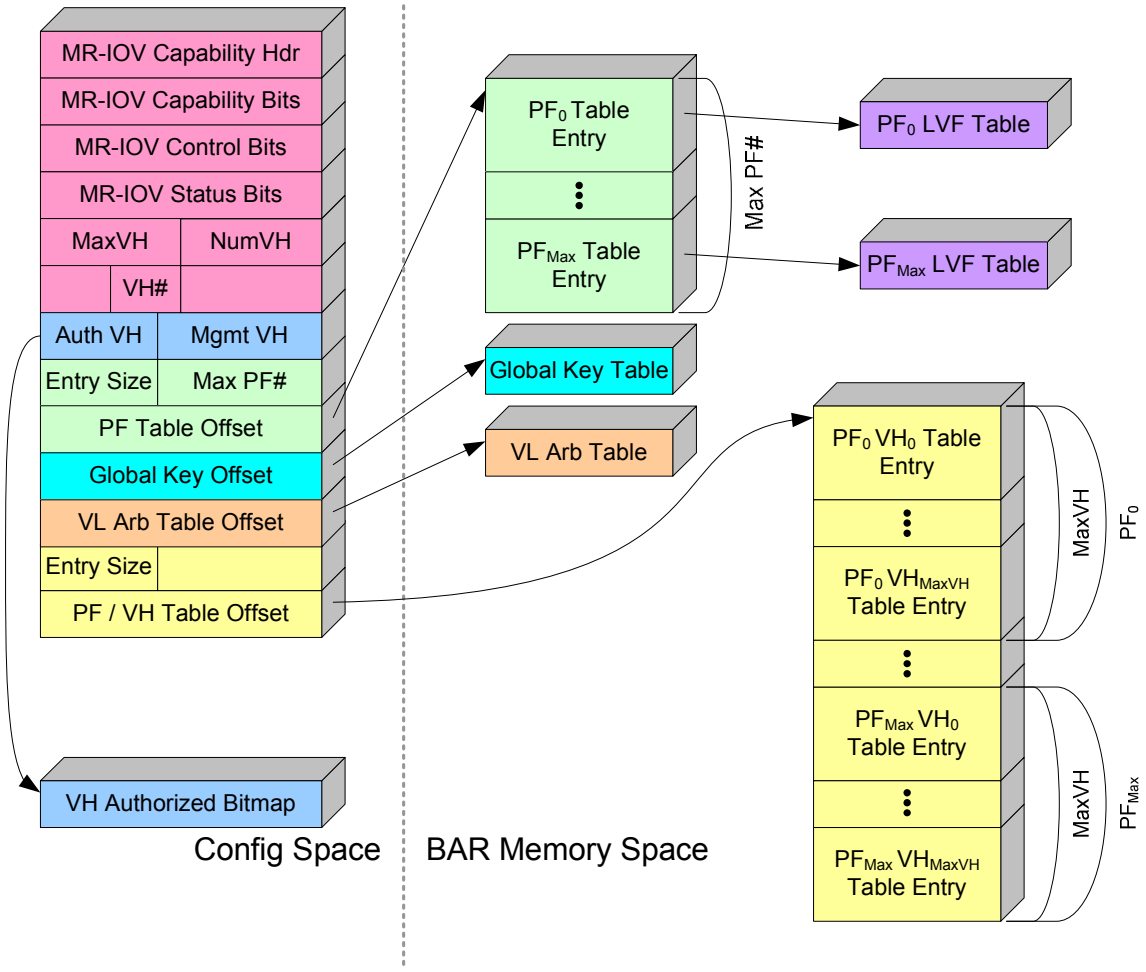


Figure 4-1: MR Device Configuration Space

4.3.1. Device MR-IOV Extended Capability

IOV adds a new, optional PCIe Extended Capability to the BF of Devices that support MR-IOV. This capability is used by MR-PCIM to determine if an EP's Base Functions is MR-IOV capable, to enable/disable its MR-IOV features, to configure the number of VHs it supports, etc.

This new capability is located in the PCIe Extended Configuration Space (offset 256 or greater) of Physical Function 0 (PF 0).

Following is the Device MR-IOV Extended Capability structure:

<table>
<tr><th colspan="2">31</th><th>19 16</th><th>15</th><th>Byte Offset</th></tr>
<tr><td colspan="2">Next Capability Pointer</td><td>Capability Version</td><td>Extended Capability ID</td><td>00h</td></tr>
<tr><td colspan="4">MR IOV Capabilities</td><td>04h</td></tr>
<tr><td colspan="4">MR IOV Control</td><td>08h</td></tr>
<tr><td colspan="4">MR IOV Status</td><td>0Ch</td></tr>
<tr><td colspan="3">NumVH (RW)</td><td>MaxVH (RO)</td><td>10h</td></tr>
<tr><td>Authorized VH Bitmap Offset (RO)</td><td colspan="2">Reserved</td><td>Management VH (RW)</td><td>14h</td></tr>
<tr><td colspan="4">Memory Mapped Configuration Space Offset (RO)</td><td>18h</td></tr>
</table>

Figure 4-2: Device MR-IOV Capability

4.3.1.1. PCI Express Extended Capability ID (00h)

This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability.

The Extended Capability ID for the MR-IOV Capability is **[TBD]**.

4.3.1.2. Capability Version (02h)

This field is a PCI-SIG defined version number that indicates the version of the capability structure present.

Must be **[TBD]** for this version of the specification.

4.3.1.3. Next Capability Offset (02h)

This field contains the offset to the next PCI Express Extended capability structure of 000h if no other items exist in the linked list of capabilities. This offset is relative to the beginning of PCI compatible configuration space and must always be either 000h or greater than 0FFh.

4.3.1.4. MR-IOV Capabilities (04h)

Table 4-2: MR-IOV Capabilities (04h)

Bit Location	Register Description	Attributes
0	**VF Migration Hardware Present** – If this fields is set, this BF contains hardware to support VF Migration. If clear, this hardware is not present.	RO
1	**VF Map Hardware Present** – If this field is set, this BF contains a VF Map Table. If Clear, VFs are assigned to PFs in a device specific manner.	RO
2	**VL Arbitration Table Present** – If this field is set, this BF contains a VL Arbitration Table.-	RO
5:3	**Max VL** – maximum supported VL number.	RO
6	**VH Authorized** – returns 1 if the VH issuing the read is authorized to manage the Device, returns 0 otherwise	RO
31:6	**Reserved**	RO

4.3.1.5. MR IOV Control (08h)

1 VH Enable

8 VL Enable

1 Migration Supported (per VH per PF)

24 (8x3) VC → VL map (per VH per PF)

3 Extended VC Count (per VH per PF)

3 Low Priority Extended VC Count (per VH per PF)

Table 4-3: MR-IOV Control (08h)

Bit Location	Register Description	Attributes
0	**MR-IOV Enable** – set to indicate the link should operate using the MR-IOV enhanced protocol.	RW
7:1	**Reserved**	RO
15:8	**VL Enable** – set to enable the associated Virtual Link. VL 0 is always enabled so bit 8 is always set.	RW
31:16	**Reserved**.	RO

4.3.1.6. MR-IOV Status (0Ch)

1 MR Enable Busy

1 Migration Enabled (per VH per PF)

8 VC Enabled (per VH per PF)

3 VC ID (per VH per PF)

8 TC to VC map (per VF per PF)

Table 4-4: MR-IOV Status (0Ch)

Bit Location	Register Description	Attributes
11:0	**Virtual Hierarchy Number** (VHN) – returns the VH number of the reader.	RO
31:12	**Reserved**	RO

4.3.1.7. MaxVH (10h)

This field defines the maximum number of VHs the EP supports for the Base Function that contains this capability structure. This field is implemented as a read-only field.

4.3.1.8. NumVH (12h)

This field defines the number of VHs that MR-PCIM has assigned to the Base Function. The Base Function shall provide Num VHs worth of Virtual Hierarchies. This field is implemented as a read-write field.

4.3.1.9. Management VH (14h)

This register indicates which VH is considered the Management VH. The active MR-PCIM software is running in the Root Port at top of this VH's PCI Express Hierarchy.

The indicated VH is automatically authorized independent of the state of the corresponding VH Authorization Bitmap entry.

This register defaults to 0.

4.3.1.10. VH Authorization Bitmap Offset (17h)

This register contains the zero-based offset of the start of the VH Authorization Bitmap in DQWORDS (16 bytes) from the base address of the MR-IOV Capability structure.

The VH Authorization Bitmap contains MaxVH bits. If the corresponding bit is set or the VH is the Management VH, the associated VH is Authorized, otherwise it is non-Authorized.

Only transactions from Authorized VH manage the Device.

The VH Authorization Bitmap defaults to bit 0 set and all other bits clear.

Note: The bitmap entry for the Management VH is read/write but its value does not affect authorization. If the "old" Management VH is to remain authorized after a transition, software should set this entry before changing Management VH to ensure that such authorization remains seamless.

4.3.1.11. Memory Mapped Configuration Space Offset (18h)

This field holds the BAR relative pointer to the remaining configuration fields that are contained in the BF's Memory Mapped Address Space. This field is implemented as a read-only field where the three lowest order bits define the BAR to use and the remaining bits indicate the offset.

4.3.2. Memory Mapped Configuration Space

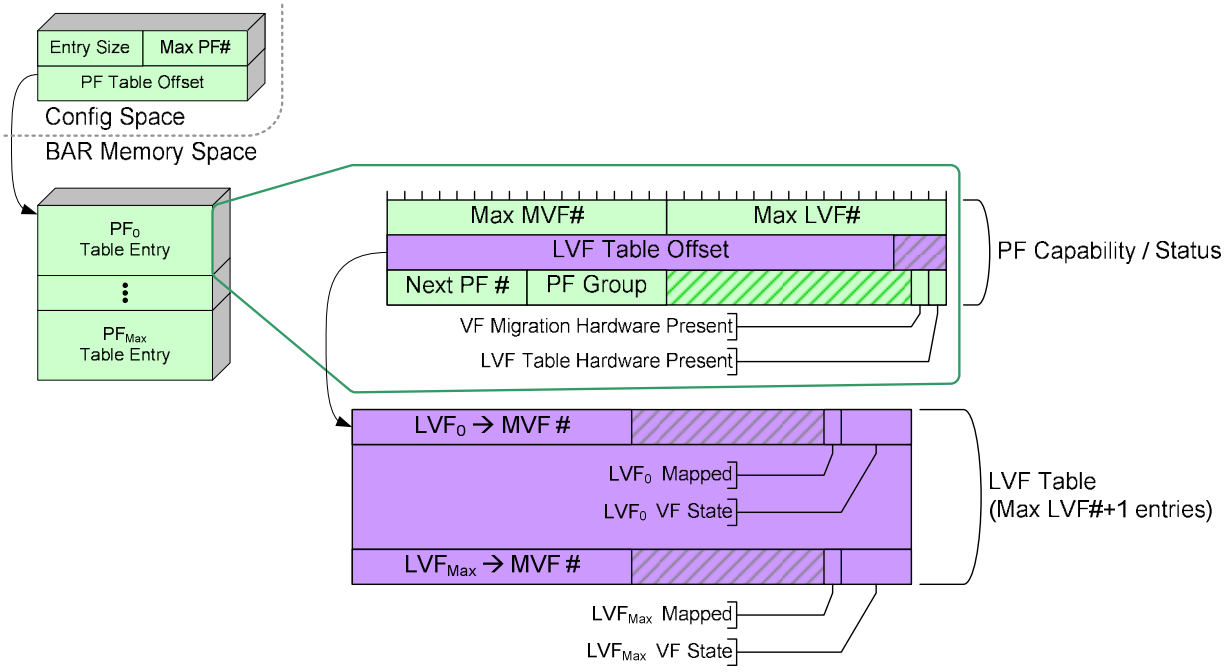


Figure 4-3: MR Device PF Table

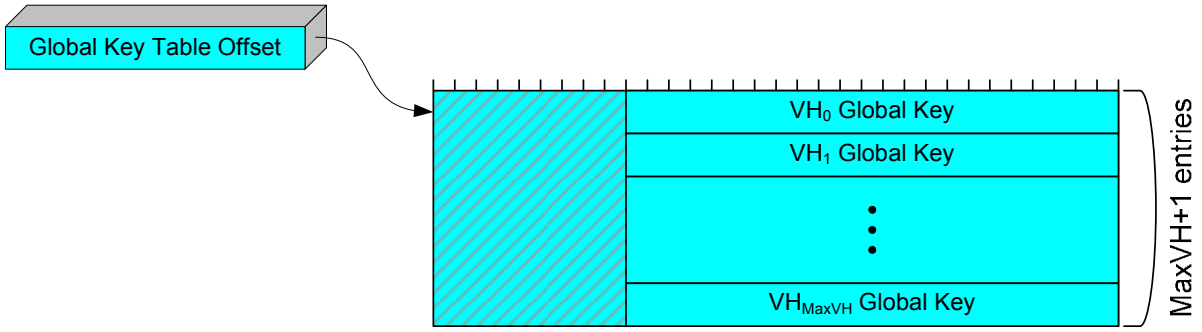


Figure 4-4: MR Device Global Key Table

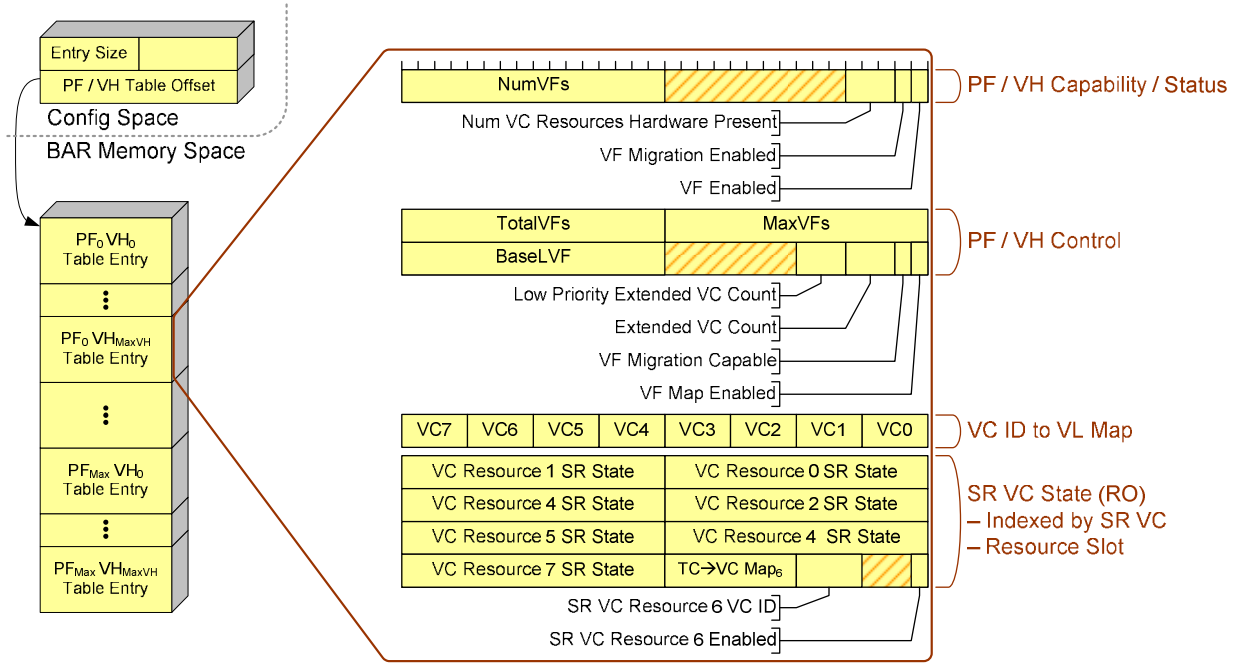


Figure 4-5: MR Device PF / VH Table

The memory mapped configuration space is located within the memory space aperture of BF and is referenced by the TBD CROSS REF.

The memory mapped configuration space fields shall provide the following capabilities:

- The ability for reusing a VF within a VH. That is, a capability will be provided that allows a VF that is in use by a SI to be removed from use by that SI and then associated with another SI.
- The ability to migrate a VF from one VH to a different VH without affecting other VFs on the EP.
- The ability to migrate a VF that has a persistent identity from one VH to a different VH.

- The ability for labeling a VF such that its identity can be made visible through PCIe capabilities when the VF is being migrated from one VH to a different VH.

Following is the proposed MR-IOV memory mapped configuration space that supports IOV. Note that the size of these fields has not been determined.

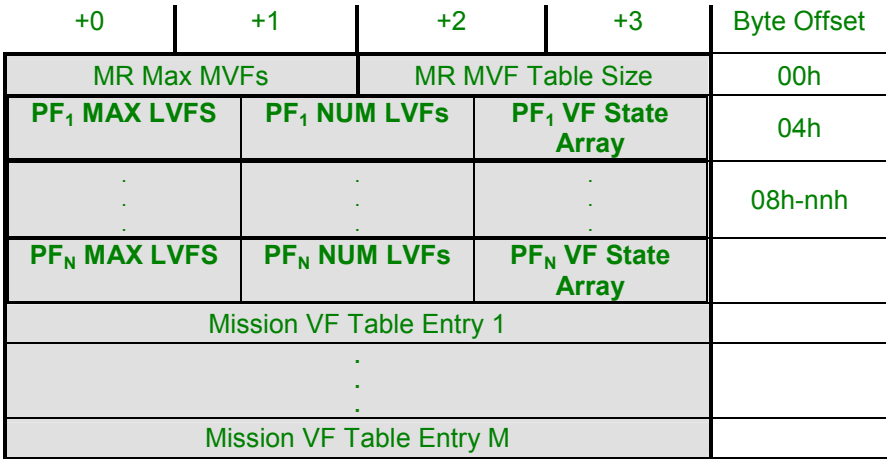


Figure 4-6: Multi-Root IO Virtualization Memory Mapped Configuration Space

4.3.2.1. MR Max MVFs

This RO field holds the maximum number of MVFs that are supported by the EP.

4.3.2.2. MR MVF Table Size

This RO field holds the number of entries in the MVF Table.

4.3.2.3. PF_N Max LVFs

This RW field holds the number of LVF slots the MR-PCIM makes available to PF_N.

4.3.2.4. PF_N Num LVFs

This RO field holds the number of LVF slots used by PF_N.

4.3.2.5. PF_N VF State Array

This RO field holds the number of state of each LVF slot used by PF_N.

4.3.2.6. The MVF Table

The Mission Virtual Function (MVF) Number serves to uniquely identify the underlying resources used by a logical VF. For example, the configuration space resources associated with a logical function reside in the MVF. An LVF is associated with a MVF through the MVF Table.

MR-PCIM assigns each PF a PF Max LVFs worth of MVF Table entries. The MVF Table entries indicate whether they are associated with an MVF.

4.3.2.6.1. MVF Table Entry M

If an MVF Table entry has an unsigned integer value greater than zero, the entry indicates the corresponding MVF. For example, if an entry has a value of 0x01h then it is associated with MVF 1. If an MVF table entry has a value of 0, then the associated LVF does not an MVF assigned to it. This field is a RW field.

4.3.3. The VF States (or MR-PCIM IOV Resource Initialization)

This section describes how the VF resources are allocated and tracked by the MR-PCIM.

The MR-PCIM begins by discovering an EP's MR IOV capabilities and reading those from the BF configuration space. MR-PCIM will read the MR-IOV Capabilities to determine that the EP is using MR-PCIM-based VF to PF mapping. MR-PCIM will read the Max VHs and Max VFs from the IOV Extended Configuration Space. The MR-PCIM will then populate the MVF Table with MVFs. Entries that are populated have MVFs associated with them. When MR-PCIM finishes configuring the EP's MR-IOV capabilities (by setting the MR-IOV enable bit), the SR-PCIM will get a PF's MVF's worth of Max VF's. SR-PCIM can then read its MVF Table segment to discover how many of those LVFs have MVFs assigned to them.

The MVF Table entry population is dictated by the states a VF goes through. During the lifetime of a MRA EP a VF can go through three possible states:

- Unassigned - The initial state of the table entry when the MR-PCIM first discovers the EP
- Assigned - The state of the table entry when MR-PCIM has created an entry in the MVF table in the process of assigning the VF to the SR-PCIM.
- In Use - The state when SR-PCIM has associated the VF with a SI.

When the MR-PCIM first discovers the EP and controls the VF state, the VF is considered to be in an "unassigned" state. At this point the appropriate entry in the MVF state table is 0. When an entry is in this state and SR-PCIM attempts to read the associated LVF's configuration space it will read a value of zero. This entry in the MVF Table allows for easy identification in the case of migration of the VF.

When the MR-PCIM assigns the VF to the SR-PCIM it marks this transition by writing the MVF number into an entry in the MVF table and a 0 into the associated VF State Array bit. An entry that is in this state can be discovered by SR-PCIM reading non-zero values in the associated LVF's configuration space. At this point the VF is considered "assigned".

Once the VF is assigned the SR-PCIM can relinquish control of the VF, assign it to an SI to use, or leave it in its assigned state. If the SR-PCIM relinquishes control, the VF reverts to its "unassigned" state and the MVF Table entry reverts to having 0 in its entry. If instead, the SR-PCIM associates the VF with a SI, the VF is considered to be "in-use" and the respective entry in the PF_N VF State Array is set to 1. Once this transition is complete the MVF/LVF is assigned to an SI.

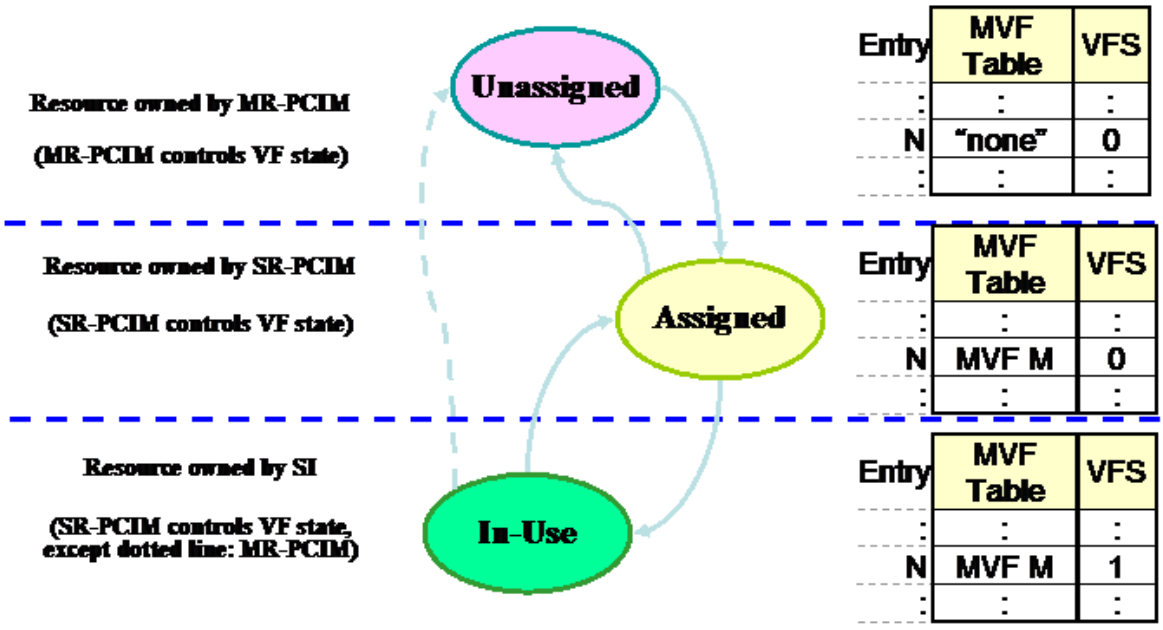


Figure 4-7: VF State Table Diagram

VF State Diagram Overview

This section will add more detail to the states with some examples.

4.3.4. Capabilities (physical device stuff, etc)

4.3.5. Virtual Power Management Controller

This section will describe registers used to manage Virtual Power State of each PF. See section 7.2 for details.

4.3.6. Actual Power Management Controller

This section will describe registers used to manage Actual Power State of the Device. See section 7.3 for details.

4.4. Switch Configuration Space

For managing MR-IOV Switches, MR-PCIM must use the Switch Type 1 Headers to manage the Switch. Configuration controls are associated with:

- Entire component
- Physical Port of the component
- Virtual Switch within the component
- PCI-to-PCI Bridges within each Virtual Switch

Accordingly, there are four main tables provided by the Switch. These tables are located using the MR-IOV Capability block located in the Type 1 Configuration Space of upstream P2P Bridge(s). Some "main" tables also point to a few "minor" tables as well. An overview of the tables is show in Figure 4-8.

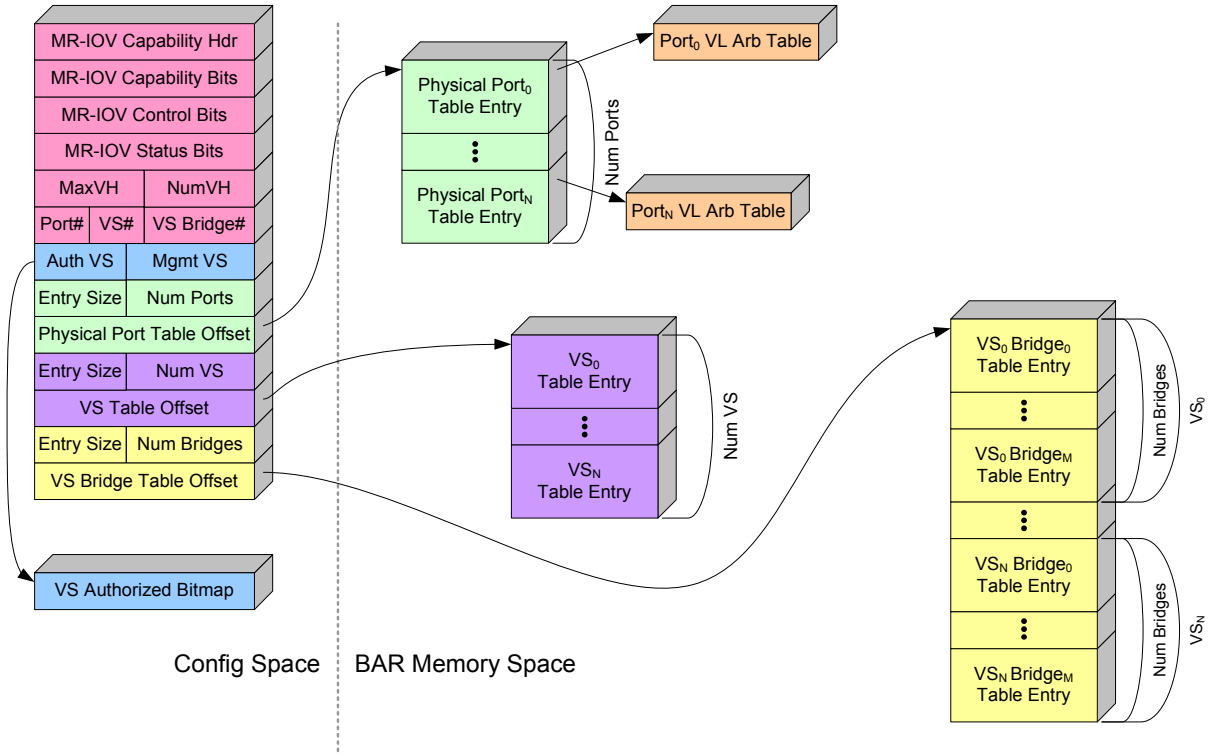


Figure 4-8: Switch Mapping Tables

- The Switch MR-IOV Capability contains the controls that apply to the entire Switch. It also contains BAR relative offsets to tables located in BAR Memory Space and associated table size information.
- The Physical Port Table contains an entry for every Port on the Switch. The table may be sparse (i.e. contain unused table entries) for hardware implementation flexibility. The Port table points to the Global Key Table and the optional VL Arbitration Table (see TBD Congestion Management chapter for VL Arb Table details).
- The VS Table contains an entry for every Virtual Switch in the MR Switch. The table may be sparse for hardware implementation flexibility.
- The VS Bridge Table contains an entry for every PCI-to-PCI bridge in a Virtual Switch. The first entry corresponds to the upstream P2P Bridge and the remaining entries are the downstream bridge(s). This table may also be sparse but the first entry must be present. This table is unused and its fields are undefined if the associated VS Table entry is not present.

These tables must be visible in the Upstream P2P Bridge of all authorized VHs as. These tables may be present in a "dormant" state the Upstream P2P Bridges of non-authorized VHs. The exact meaning of "dormant" will be defined in a later version of this specification but will, at a minimum, prevent software running in non-authorized VHs from affecting the MR operation of the Switch.

In addition, the MR-IOV Capabilities structure must be present in the VH 0 Upstream P2P Bridge that is attached to MR aware Root Ports. For non-authorized VHs, this may be a subset consisting of only the Configuration Space fields and containing none of the in-memory tables.

4.4.1. Switch MR-IOV Extended Capability

Following is the Switch MR-IOV capability structure:

31	19 16	15	Byte Offset
Next Capability Pointer	Capability Version	Switch MR-IOV Extended Capability ID	00h
MR IOV Capabilities (RO)			04h
MR IOV Control			08h
MR IOV Status			0Ch
NumVH (RW)		MaxVH (RO)	10h
Authorized VS Bitmap Offset (RO)	Reserved	Management VS (RW)	14h
Port # (RO)	VS # (RO)	VS Bridge # (RO)	18h
Port Entry Size (RO)		Num Port Entries (RO)	1Ch
Port Table Offset (RO)		BIR	20h
VS Entry Size (RO)		Num VS Entries (RO)	24h
VS Table Offset (RO)		BIR	28h
VS Bridge Entry Size (RO)		Num VS Bridge Entries per VS (RO)	2Ch
VS Bridge Table Offset (RO)		BIR	30h

Figure 4-9: Switch MR-IOV Capability

The fields in the Switch MR-IOV Capability are:

4.4.1.1. Switch MR-IOV Extended Capability Header (00h)

Standard PCI Express Extended Configuration Header. Capability ID is TBD. Version is 1h.

4.4.1.2. Switch MR-IOV Capability (04h)

Authorized Bit (1 bit)

MR-IOV Enable Bit (1 bit)

Interrupt MSI Vector# (11 bit)

4.4.1.3. Switch MR-IOV Control (08h)

Port Interrupt Enable (1 bit)

VS Interrupt Enable (1 bit)

MR Switch Number (TBD Bits), used by MR-PCIM during enumeration to detect topology loops.

4.4.1.4. Switch MR-IOV Status (0Ch)

Port Interrupt Status (1 bit RW1C)

VS Interrupt Status (1 bit RW1C)

4.4.1.5. MaxVH (10h)

This read only register indicates the number of VHs being offered by an upstream Switch Port. This is similar to the MaxVH field in a device. The value is controlled by MR-PCIM using the MaxVH field in the Physical Port Table.

4.4.1.6. NumVH (12h)

This register indicates how many of the offered VHs are being used by software upstream of the upstream Port. This is similar to the NumVH field in a device. The value set is readable by MR-PCIM using the NumVH field in the Physical Port Table.

4.4.1.7. Management VS (14h)

This register indicates which VS is considered the primary Management VH. The active MR-PCIM software is running in the Root Port at top of this VH's PCI Express Hierarchy.

The indicated VS is automatically authorized independent of the state of the corresponding VS Authorization Bitmap entry.

4.4.1.8. VS Authorization Bitmap Offset (17h)

This register contains the zero-based offset of the start of the VS Authorization Bitmap in DQWORDS (16 bytes) from the base address of the MR-IOV Capability structure.

The VS Authorization Bitmap contains Num_VS_Table_Entries bits. If the corresponding bit is set or the VS is the Management VS, the associated VS is Authorized, otherwise it is non-Authorized.

Only transactions from Authorized VS are allowed to access memory space MR-IOV tables of this Switch.

Note: The bitmap entry for the Management VS is read/write but its value does not affect authorization. If the "old" Management VS is to remain authorized after a transition, software should set this entry before changing Management VS to ensure that such authorization remains seamless.

4.4.1.9. Port# / VS # / VS Bridge# (18h)

This register returns the Port#, VS# and VS Bridge# of the associated PCI-to-PCI bridge. These values can be used by MR-PCIM to locate corresponding the Physical Port Table, VS Table and VS Bridge Table entries.

This register is read only and is implemented in each Type 1 Configuration Space. Values returned for VS# and VS Bridge # corresponding to Ports that to not have a VS and Bridge mapped is TBD.

4.4.1.10. Physical Port Entry Size / Num Physical Port Entries (1Ch)

This register returns the number of entries and the size of each entry in the Physical Port Table. The total table size (in bytes) is:

Num_Physical_Port_Entries * Physical_Port_Entry_Size

Subsequent versions of this specification may increase the entry size.

4.4.1.11. Physical Port Entry Table Offset (20h)

This register returns a BAR relative offset to the first entry in the Physical Port Table. The low 3 bits contain a BIR field that indicates a BAR and the upper 29 bits contain a BAR relative offset to the table entry. The offset is in QWORDs.

4.4.1.12. VS Table Entry Size / Num VS Table Entries (24h)

This register returns the number of entries and the size of each entry in the VS Table. The total table size (in bytes) is:

Num_VS_Table_Entries * VS_Table_Entry_Size

Subsequent versions of this specification may increase the entry size.

4.4.1.13. VS Table Offset (28h)

This register returns a BAR relative offset to the first entry in the VS Table. The low 3 bits contain a BIR field that indicates a BAR and the upper 29 bits contain a BAR relative offset to the table entry. The offset is in QWORDs.

4.4.1.14. VS Bridge Table Entry Size / Num VS Bridge Table Entries per VS (2Ch)

This register returns the size of each entry in the VS Bridge Table and the number of VS Bridge Table entries per VS. The total size (in bytes) of the VS Bridge Table is:

Num_VS_Bridge_Table_Entries * Num_VS_Table_Entries * VS_Bridge_Table_Entry_Size

Subsequent versions of this specification may increase the entry size.

Note: VS Bridge Table Entries per VS reflects the size of the table and thus is the maximum size of any VS. Not all VB Bridge Table entries need be present (see Section 4.4.4.1).

4.4.1.15. VS Bridge Table Offset (30h)

This register returns a BAR relative offset to the first entry in the VS Bridge Table. The low 3 bits contain a BIR field that indicates a BAR and the upper 29 bits contain a BAR relative offset to the table entry. The offset is in QWORDs.

The VS Bridge Table associated with bridge 0 of VS *N* immediately follows the last bridge table entry associated with VS *N-1*.

4.4.2. Switch Physical Port Table

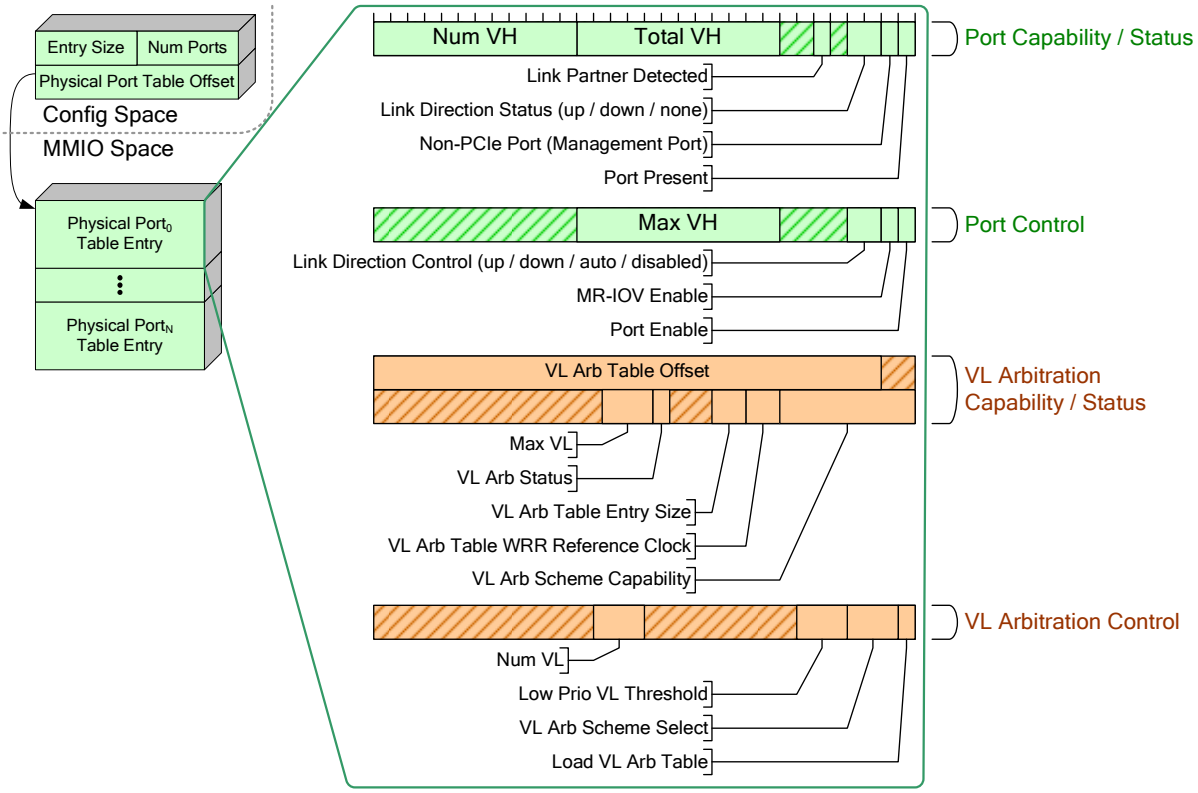


Figure 4-10: Switch Physical Port Table

4.4.2.1. Port Present (RO)

Indicates the Port hardware is present and the rest of the table entry is usable. The value may be altered in a Vendor Specific fashion to, for example, to support reconfigurable Ports with variable widths.

4.4.2.2. Non-PCIe Port (RO)

Indicates that the Port hardware is a Vendor Specific non-PCIe port used for management. Certain bits in the Physical Port Entry are no longer meaningful. TODO: Specify these bits.

4.4.2.3. Port Enable (RW)

Set by MR-PCIM to indicate that it wishes to use the Port. Impact is TBD.

4.4.2.4. MR-IOV Enable (RW)

Set by MR-PCIM to indicate that it wishes to use the MR-IOV enhanced protocol on the Port.

4.4.2.5. Link Direction Status (RO)

Set to Indicate the phy training link direction (upstream, downstream and link-down / non-trained).

4.4.2.6. Link Partner Detected (RO)

Set to indicate that a PCI Express component was detected at the remote end of the one or more lanes of the link.

4.4.2.7. Link Direction Control (RW)

Set to control the phy link training direction. Values include up stream, downstream, automatic and disable link.

4.4.2.8. Total VH Implemented (RO)

Number of VHs implemented in Silicon.

4.4.2.9. MaxVH (RW)

Number of VHs offered to software running in the VH. Corresponds to the MaxVH read only field in the MR-IOV Capability.

4.4.2.10. NumVH (RO)

Number of VHs used by software running in the VH. Corresponds to the NumVH read write field in the MR-IOV Capability.

4.4.2.11. VL Arbitration fields

These fields are similar to the VC arbitration fields in the VC Extended Capability. VC Arb Offset is a BAR relative pointer.

TODO: Describe these fields.

4.4.3. Switch VS Table

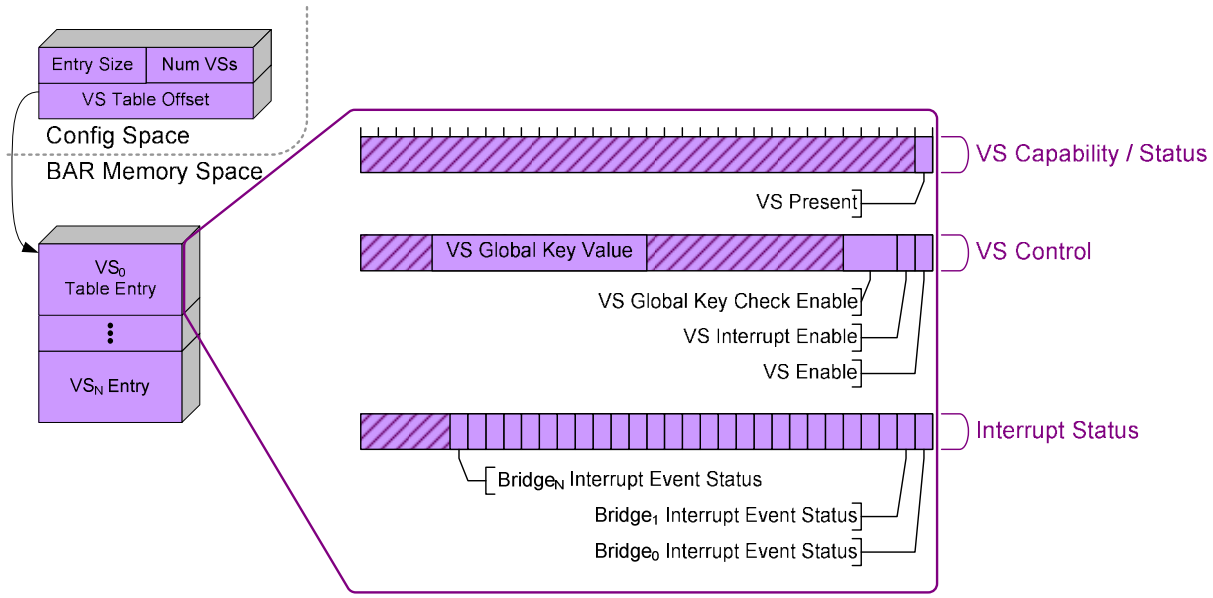


Figure 4-11: VS Table

4.4.3.1. VS Present

Read only bit indicating that the VS is implemented. When VS Enable is zero, this bit may be controlled in a Vendor Specific mechanism to allow for flexible silicon implementation.

4.4.3.2. VS Enable

If set, indicates that MR-PCIM is using this VS. When set, the capabilities of the VS may not change. When clear, Vendor Specific mechanisms may change the capabilities offered by this VS. If the Switch does not support changing VS capabilities, this bit may be read only with the value one.

4.4.3.3. VS Interrupt Enable

Allows the Bridge *N* Interrupt Status bits to cause a VS Interrupt.

4.4.3.4. VS Global Key Check Enable

Enables checking of the Global Key in the TLP Prefix against the Global Key Value for TLPs associated with the VS.

This is a three bit field that enables checks at various points in TLP processing.

- Bit 0 enables the terminating check. This validates TLPs addressing the VS. Every VS must implement this bit.
- Bit 1 enables the entering check. This validates forwarded TLPs as they are received by the Switch. Implementing this check is optional. If not implemented, this bit is read only zero.
- Bit 2 enables the exiting check. This validates forwarded TLPs as they are transmitted by the Switch. Implementing this check is optional. If not implemented, this bit is read only zero.

A Switch may implement the bit 1 check, the bit 2 check, both checks or neither check. When implemented, a check must be implemented in every VS.

4.4.3.5. VS Global Key Value

Expected Global Key Value to for TLPs associated with the VS.

This value is inserted in TLPs originated by the VS.

This value is inserted in TLPs entering the VS from a non-MR Enabled Link.

If enabled, Global Keys for TLPs associated with the VS are checked against this value.

4.4.3.6. $Bridge_N$ Interrupt Status

Indicates that Bridge *N* associated with this VS is requesting an interrupt. This bit is set if the interrupt condition exists in the VS Bridge and if the interrupt is enabled in the associated VS Bridge Table entry.

4.4.4. Switch VS Bridge Table

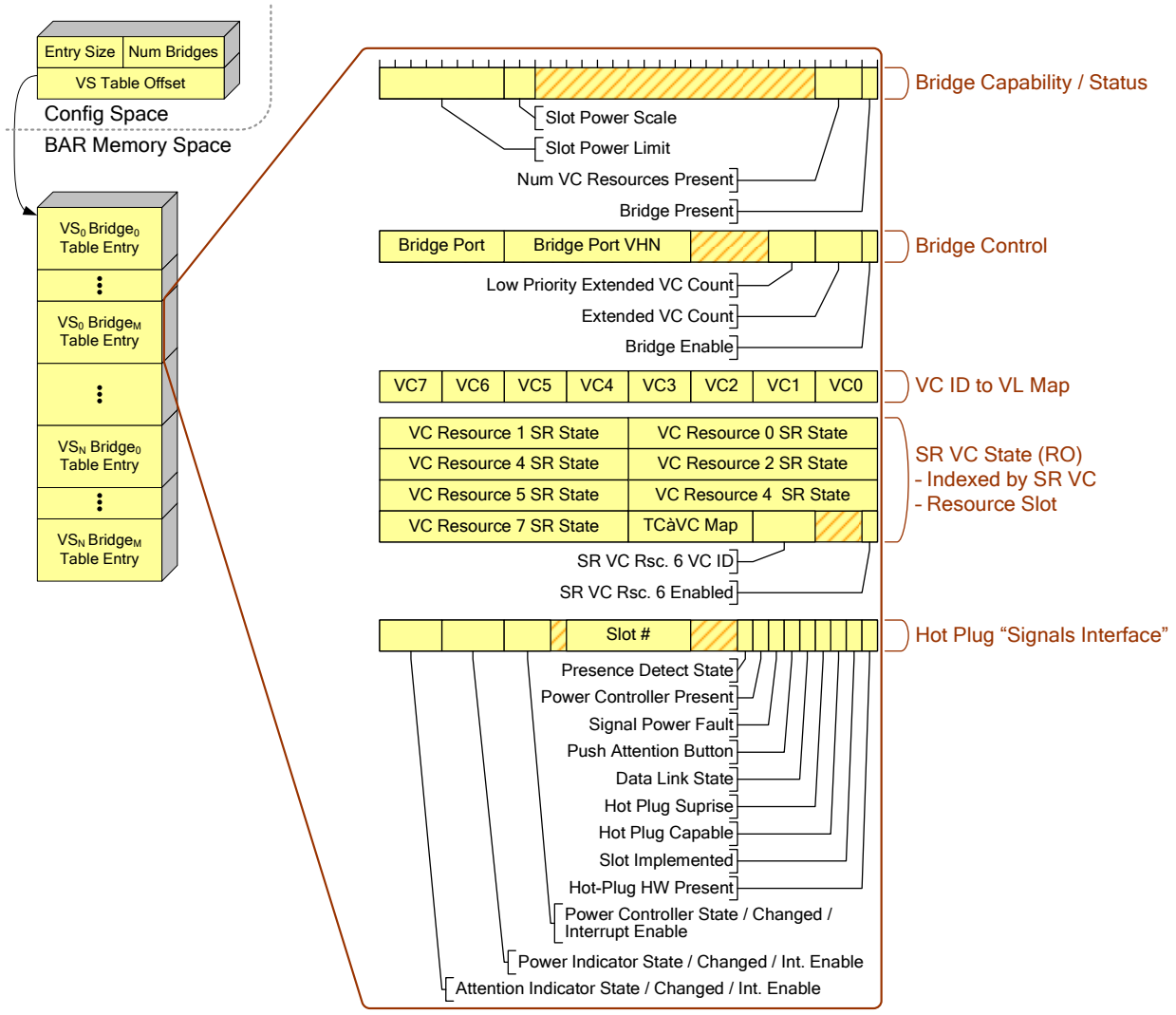


Figure 4-12: VS Bridge Table

VS Bridge Table Entry 0 corresponds to the upstream bridge in the VH. This bridge is always present and enabled.

VS Bridge Table Entry 1 corresponds to the downstream P2P Bridge located at Device 0 in the VH.

VS Bridge Table Entry *M* corresponds to the downstream P2P Bridge located at Device *M-1* in the VH.

4.4.4.1. Bridge Present (RO)

Set if the hardware supports a bridge in this slot. The VS Bridge Table has the same number of Table Entries associated with each VS. This bit allows the table to be sparse and to be populated in a Vendor Specific manner. This bit is not allowed to change if VS Enabled is set.

The Upstream Bridge is always present in any enabled VS. The first downstream bridge (table entry 1 representing device 0) is also always present.

When VS Enable is clear, Vendor Specific mechanisms may change the value of this field.

4.4.4.2. Bridge Enable (RW)

If set, the associated P2P Bridge is visible in the associated VS and the associated Type 1 Configuration Header is accessible.

If clear, the associated P2P Bridge is disabled and the associated Type 1 Configuration Space is not accessible to software in the VS.

Clearing Bridge Enable for an Upstream VS Bridge Table entry blocks access to bridges below the upstream bridge and thus has the effect of hiding the entire VS from the view of software running in the VH.

Enabling a VS is a two step process. First enable the VS. This ensures that Vendor Specific mechanisms will not alter the Bridge Present bits during the rest of this process. Then configure and enable specific bridges in the VS starting with downstream bridges and finishing with the upstream bridge.

If Bridge Present is clear this field is undefined.

The default value of this field is Vendor Specific.

4.4.4.3. Num VC Resources Hardware Present (RO)

Indicates the index of the last VC Resource array structure implemented in the Type 1 Configuration Header associated with this Bridge in the VH. The value 0 indicates one VC Resource is provided. The value 7 indicates that all 8 VC Resources are provided.

This value indicates the number that the hardware implements. MR-PCIM software may offer a lower number to the VH by setting Extended VC Count.

When VS Enable is clear, Vendor Specific mechanisms may change the value of this field.

4.4.4.4. Extended VC Count (RW)

MR-PCIM must configure this value with the number of VC Resources that are offered to software operating in the VH. This value will be available as Extended VC Count in the VC Capability of the associated Type 1 Configuration Header.

This value must be less than or equal to Num VC Resources Hardware Present. Changing this value while Bridge Enable is set is undefined. The default value of this field is Vendor Specific.

4.4.4.5. Low Priority Extended VC Count (RW)

MR-PCIM must configure this value with the value to be provided to software operating in the VH. This value has no hardware effect. This field's purpose is to inform software running in the VH of the relative priority of certain VCs.

This value must be less than or equal to the value set for Extended VC Count. Changing this value while Bridge Enable is set is undefined. The default value of this field is Vendor Specific.

4.4.4.6. Slot Power Limit Value (RO)

This field contains the value written to the PCI Express Capability Slot Power Limit Value register by software in the VH. There is no other effect.

If Hot-Plug support is not provided or is not enabled for this bridge entry (i.e. Slot Implemented is 0), this field is undefined.

4.4.4.7. Slot Power Limit Scale (RO)

This field contains the value written to the PCI Express Capability Slot Power Limit Scale register by software in the VH. There is no other effect.

If Hot-Plug support is not provided or is not enabled for this bridge entry (i.e. Slot Implemented is 0), this field is undefined.

4.4.4.8. Bridge VHN (RW)

This field indicates the Port VHN associated with this P2P Bridge. This value must be less than the number of VHs implemented by the Port contained in Bridge Port. The default value of this field is Vendor Specific.

4.4.4.9. Bridge Port (RW)

This field indicates the Port associated with this P2P Bridge. This value must correspond to an enabled Physical Port. The default value of this field is Vendor Specific.

4.4.4.10. VCn to VL Map (RW)

These fields contain the Virtual Link to be used for traffic out this P2P Bridge for the indicated VC.

Implementing the entry for VC 0 is mandatory. Implementing the other entries is optional for the case where only one VC is supported (as indicated by Num VC Resources Hardware Present being 0).

These fields may not be changed while Bridge Enable is set. The default value of these fields is Vendor Specific.

4.4.4.11. VC Resource n fields (RO)

These fields return data from the VC Capability of the associated Type 1 Configuration Header. They allow MR-PCIM software to track the enabling and mapping of VCs with each VH.

VC Resource fields for resource numbers above Num VC Resource Hardware Present are not implemented and return 0 when read. VC Resource fields for resource numbers above Extended VC Count are undefined.

The default value of these fields corresponds to the value specified in the PCI Express specification for the VC Capability.

4.4.4.11.1. VC Resource n: VC Enabled (RO)

This field tracks the VC Enabled bit set by SR software operating in the VH. Per the PCI Express specification, VC Resource 0 is always enabled and thus VC Enabled for VC Resource 0 is always set.

4.4.4.11.2. VC Resource N: VC ID (RO)

This field tracks the VC ID field set by SR software operating in the VH. Per the PCI Express specification, VC ID for VC Resource 0 is always 0.

4.4.4.11.3. VC Resource N: TC to VC Map (RO)

This field tracks the TC to VC Map field set by SR software operating in the VH. Per the PCI Express specification, bit 0 of this field is fixed and the remaining bits may be set by software.

4.4.4.12. Virtual Hot-Plug Control Bits

These bits form the "signals interface" of the Virtual Hot-Plug controller. See Section 6 Hot Plug for details.

Hot-Plug controller hardware is optional. If not present, the Slot Implemented bit is Read Only zero and the remaining bits in this word are undefined.

4.4.4.12.1. Hot-Plug Hardware Present (RO)

This field indicates whether the P2P Bridge supports Hot-Plug. If clear, Hot-Plug hardware is not present and the remainder of the "signals interface" bits are read only zero.

When VS Enable is clear, Vendor Specific mechanisms may change the value of this field.

4.4.4.12.2. Slot Implemented (RW)

This field indicates to SR software whether the P2P Bridge contains Hot-Plug support. It is echoed to SR software as the Slot Implemented bit in the PCI Express Capabilities field.

4.4.4.12.3. Physical Slot Number (RW)

This field indicates the chassis slot number. It is echoed to SR software in the PCI Express Capabilities field. The default value of this field is Vendor Specific.

4.4.4.12.4. Hot Plug Capable (RW)

This field indicates that the slot is Hot-Plug Capable. It is echoed to SR software in the Slot Capabilities field of the PCI Express Capabilities. The default value of this field is Vendor Specific.

4.4.4.12.5. Hot Plug Surprise (RW)

This field indicates that the slot support Surprise Hot-Plug events. It is echoed to SR software in the Slot Capabilities field of the PCI Express Capabilities. The default value of this field is Vendor Specific.

4.4.4.12.6. Data Link State (RW)

This field indicates that state of the Virtual Data Link. It is echoed to SR software in the Slot Status field of the PCI Express Capabilities. Changing this field causes a Data Link State Change event in the virtual Hot-Plug controller. The default value of this field is Vendor Specific.

4.4.4.12.7. Attention Indicator State (RO)

This field indicates the state of the Attention Indicator. It echoes the value set by SR software in the Slot Control field of the PCI Express Capabilities.

4.4.4.12.8. Attention Indicator State Change (RW1C)

This field indicates to MR-PCIM that the Attention Indicator State has been changed by SR software. MR-PCIM can clear this bit by writing a 1. Writes of 0 have no effect.

4.4.4.12.9. Attention Indicator Change Interrupt Enable (RW)

If set, assertions of the Attention Indicator State Change bit can trigger an MR-PCIM interrupt. The Attention Indicator State Change bit is not affected by the setting of this bit. An interrupt is triggered when on any 0→1 transition of Attention Indicator State Change anded with this bit.

4.4.4.12.10. Power Indicator State (RO)

This field indicates the state of the Power Indicator. It echoes the value set by SR software in the Slot Control field of the PCI Express Capabilities.

4.4.4.12.11. Power Indicator State Change (RW1C)

This field indicates to MR-PCIM that the Power Indicator State has been changed by SR software. MR-PCIM can clear this bit by writing a 1. Writes of 0 have no effect.

4.4.4.12.12. Power Indicator Change Interrupt Enable (RW)

If set, assertions of the Power Indicator State Change bit can trigger an MR-PCIM interrupt. The Power Indicator State Change bit is not affected by the setting of this bit. An interrupt is triggered when on any 0→1 transition of Power Indicator State Change anded with this bit.

4.4.4.12.13. Power Controller State (RO)

This field indicates the state of the Power Controller. It echoes the value set by SR software in the Slot Control field of the PCI Express Capabilities.

4.4.4.12.14. Power Controller State Change (RW1C)

This field indicates to MR-PCIM that the Power Controller State has been changed by SR software. MR-PCIM can clear this bit by writing a 1. Writes of 0 have no effect.

4.4.4.12.15. Power Controller Change Interrupt Enable (RW)

If set, assertions of the Power Controller State Change bit can trigger an MR-PCIM interrupt. The Power Controller State Change bit is not affected by the setting of this bit. An interrupt is triggered when on any 0→1 transition of Power Controller State Change anded with this bit.

4.4.4.12.16. Indicate Power Fault (RW)

Writing 1 to this field simulates a Power Fault condition. This fault is echoed to SR software in the Slot Status field of the PCI Express Capabilities.

4.4.4.12.17. Presence Detect State (RW)

This field indicates that the virtual slot has a card in it. It is echoed to SR software in the Slot Status field of the PCI Express Capabilities. Transitions if this bit can cause a Present Detect interrupt in the VH.

4.4.4.12.18. Press Attention Button (RW)

Writing 1 to this field simulates a press of the virtual Attention Button associated with the virtual slot. This is echoed to SR software in the Slot Status field of the PCI Express Capabilities.

5

5. Error Handling

5.1. Requirements

- The specification shall report errors isolated to a single VH within that VH and using PCIe semantics.
- The specification shall define a mechanism so that MR PCIM can be informed of any error in the system.
- The specification shall not define mechanisms to ensure that error messages are reliably delivered to MR PCIM during MR PCIM failover.
- The specification shall support the same mechanisms of error handling and reporting as specified in the *PCI Express Base Specification*. It shall allow devices to optionally implement Advanced Error Reporting.
- The specification shall specify which PCIe errors are VH-specific errors and which are global errors that must be reported to one or more VHs.
- The specification shall specify new errors related to MR and how they should be handled.

5.2. PCIe Error mapping to MR

The basic rules for error detection, logging and reporting are unchanged from PCIe. The only change is which VH(s) should be affected.

Table 5-1: Physical Layer Error List

Error Name	Error Type	Detecting Agent Action (PCIe)	Detecting Agent Action (MR IOV)
Receiver Error	Correctable	*Receiver (if checking):* Send ERR_COR to Root Complex.	*Receiver:* Same as PCIe but send on all VHs not in Reset.

Table 5-2: Data Link Layer Error List

Error Name	Error Type	Detecting Agent Action (PCIe)	Detecting Agent Action (MR IOV)
Bad TLP	Correctable	*Receiver:* Send ERR_COR to Root Complex.	*Receiver:* Same as PCIe but send to all VH that are active.
Bad DLLP	Correctable	*Receiver:* Send ERR_COR to Root Complex.	*Receiver:* Same as PCIe but send on all VHs that are active.
Replay Timeout	Correctable	*Transmitter:* Send ERR_COR to Root Complex.	*Transmitter:* Same as PCIe but send on all VHs that are active
REPLAY NUM Rollover	Correctable	*Transmitter:* Send ERR_COR to Root Complex.	*Transmitter:* Same as PCIe but send on all VHs that are active
Data Link Layer Protocol Error	Uncorrectable (Fatal)	If checking, send ERR_FATAL to Root Complex.	Same as PCIe but send to all VHs that are active.

Table 5-3: Transaction Layer Error List

Error Name	Error Type	Detecting Agent Action (PCIe)	Detecting Agent Action (MV IOV)
Poisoned TLP Received	Uncorrectable (Non-Fatal)	*Receiver:* Send ERR_NONFATAL to Root Complex, or ERR_COR for the Advisory Non-Fatal Error cases Log the header of the Poisoned TLP.	*Receiver:* Same as PCIe. Error message sent only for affected VH. Log the header for the affected VH only.
ECRC Check Failed	Uncorrectable (Non-Fatal)	*Receiver (if ECRC checking supported):* Send ERR_NONFATAL to Root Complex, or ERR_COR for the Advisory Non-Fatal Error case Log the header of the TLP that encounter the ECRC error.	*Receiver (if ECRC checking supported):* Same as PCIe Error message sent only for affected VH. Log the header for the affected VH only.
Unsupported Request (UR)	Uncorrectable (Non-Fatal)	*Request Receiver:* Send ERR_NONFATAL to Root Complex, or ERR_COR for the Advisory Non-Fatal Error case Log the header of the TLP that caused the error.	*Request Receiver:* Same as PCIe Error message sent only for affected VH. Log the header for the affected VH only.
Completion Timeout	Uncorrectable (Non-Fatal)	*Requester:* Send ERR_NONFATAL to Root Complex, or ERR_COR for the Advisory Non-Fatal Error case	*Requester:* Same as PCIe Error message sent only for affected VH.
Completer Abort	Uncorrectable (Non-Fatal)	*Completer:* Send ERR_NONFATAL to Root Complex, or ERR_COR for the Advisory Non-Fatal Error case Log the header of the Request that encountered the error.	*Completer:* Same as PCIe Error message sent only for affected VH. Log the header for the affected VH only.
Unexpected Completion	Uncorrectable (Non-Fatal)	*Receiver:* Send ERR_COR to Root Complex. This is an Advisory Non-Fatal Error Log the header of the Completion that encountered the error.	*Receiver:* Same as PCIe. Error message sent only for affected VH. Log the header for the affected VH only.
Receiver Overflow	Uncorrectable (Fatal)	*Receiver (if checking):* Send ERR_FATAL to Root Complex.	*Receiver (if checking):* Same as PCIe. ???Not sure how this is impacted by new FC scheme???
Flow Control	Uncorrectable	*Receiver (if checking):*	*Receiver (if checking):*

Error Name	Error Type	Detecting Agent Action (PCIe)	Detecting Agent Action (MV IOV)
Protocol Error	(Fatal)	Send ERR_FATAL to Root Complex.	Same as PCIe. ???Not sure how this is impacted by new FC scheme???
Malformed TLP	Uncorrectable (Fatal)	*Receiver:* Send ERR_FATAL to Root Complex. Log the header of the TLP that encountered the error.	*Receiver:* Same as PCIe. Error message sent only for affected VH. Log the header for the affected VH only.

5.3. MR Errors

Table 5-4: MR Error List

Error Name	Error Type	Detecting Agent Action (PCIe)	Detecting Agent Action (MR IOV)
Invalid TLP Prefix	Correctable	N/A	*Receiver:* Silently discard TLP. Do not update any flow control state.
TLP received on VH in reset	Correctable	N/A	*Receiver:* Silently discard TLP. Do not update any flow control state.
TLP Prefix with Global Key Mismatch	Uncorrectable (Fatal)	N/A	*Receiver:* ?
TLP Prefix VL not valid for VH	Uncorrectable (Fatal)	N/A	*Receiver:* ?
Invalid VH in FC DLLP	Correctable	N/A	*Receiver:* Silently discard DLLP.
FC DLLP for VH in Reset	Correctable	N/A	*Receiver:* Silently discard DLLP.
Invalid VH Group in Reset DLLP	Correctable	N/A	*Receiver:* Silently discard DLLP.
Out of range Assert bit set in Reset DLLP	Correctable	N/A	*Receiver:* Ignore the offending Assert bit(s) and process remainder of Reset DLLP normally.

6

6. Hot Plug

6.1. Requirements

- The specification shall utilize PCIe Hot Plug mechanisms for reporting Hot Plug events to a VH.
- MR PCIM shall be responsible for controlling the physical Hot Plug structures for all MR Devices.
- MR PCIM shall be enabled to control the policy of all Hot Plug events within the system.
- MR PCIM shall be enabled to create a Hot Plug event for any device within the system.
- The specification shall support the ability to perform a Hot Remove / Hot Add of a Virtual Device within a VH.
- The specification shall support the ability to perform a Hot Remove / Hot Add of a PCIe Device within a VH.

6.2. MRA Switch

Providing Hot Plug capabilities affects areas in an MRA Switch

6.2.1. Virtual Slot Registers

The Type 1 header associated with each virtual downstream Switch Port shall contain the PCIe Slot Capabilities, Control and Status registers. These registers work exactly as in base PCIe as described in sections 7.8.9, 7.8.10 and 7.8.11. The difference is that in MR they are attached to a "virtual signals interface" instead of connecting to real buttons, lights and sensors.

Bits are implemented as shown in the tables below. Figures are extracted for reference from the *PCI Express Base Specification*.

Writing the Slot Power bits changes the register contents but has no other effects (existing software expects this to be a read/write field).

The Slot Implemented bit (located in the PCI Express Capability) is also controlled by MR-PCIM allowing MR-PCIM to disable Hot-Plug software's use of a virtual downstream P2P bridge Hot-Plug controller.

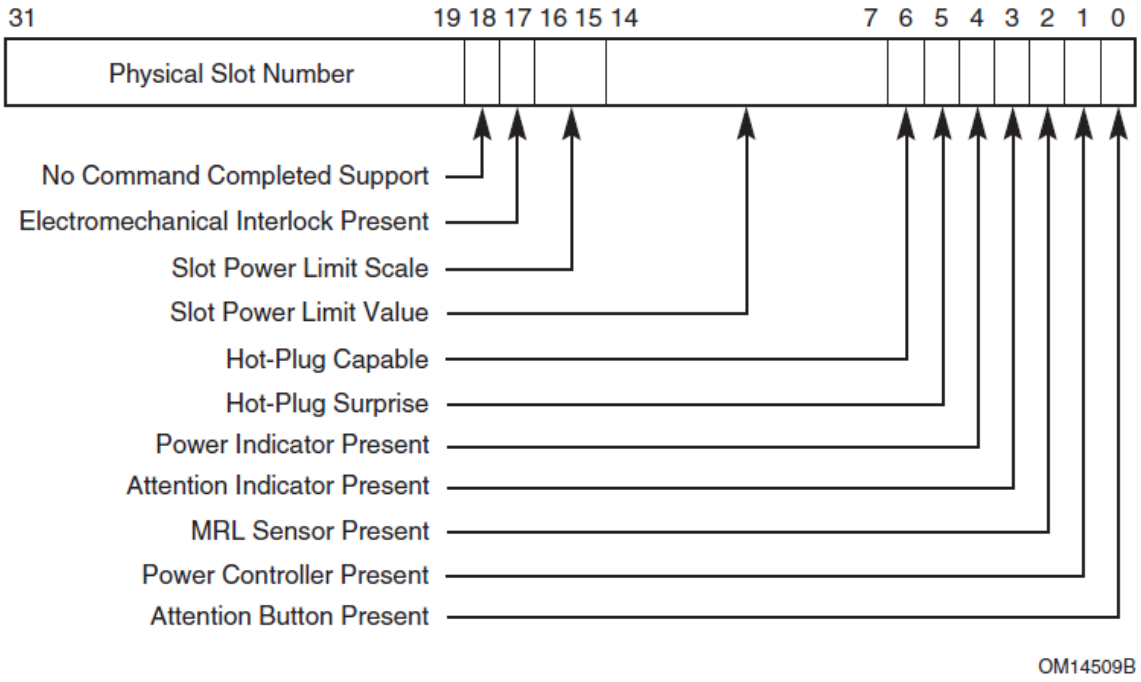


Figure 6-1: Slot Capabilities Register (PCIe Figure 7-18)

Bit Field	Implement, Zero, One, Optional
Physical Slot Number	MR-PCIM programmed value
No Command Completed Support	Zero (i.e. Implemented)
Electromechanical Interlock Present	Zero
Slot Power Limit Scale	Implemented (as a RAM, write has no effect)
Slot Power Limit Value	Implemented (as a RAM, write has no effect)
Hot Plug Capable	Implement (allow MR-PCIM to control)
Hot-Plug Surprise	Implement (allow MR-PCIM to control)
Power Indicator Present	One
Attention Indicator Present	One
MRL Sensor Present	Zero
Power Controller Present	Implement (allow MR-PCIM to control)
Attention Button Present	One

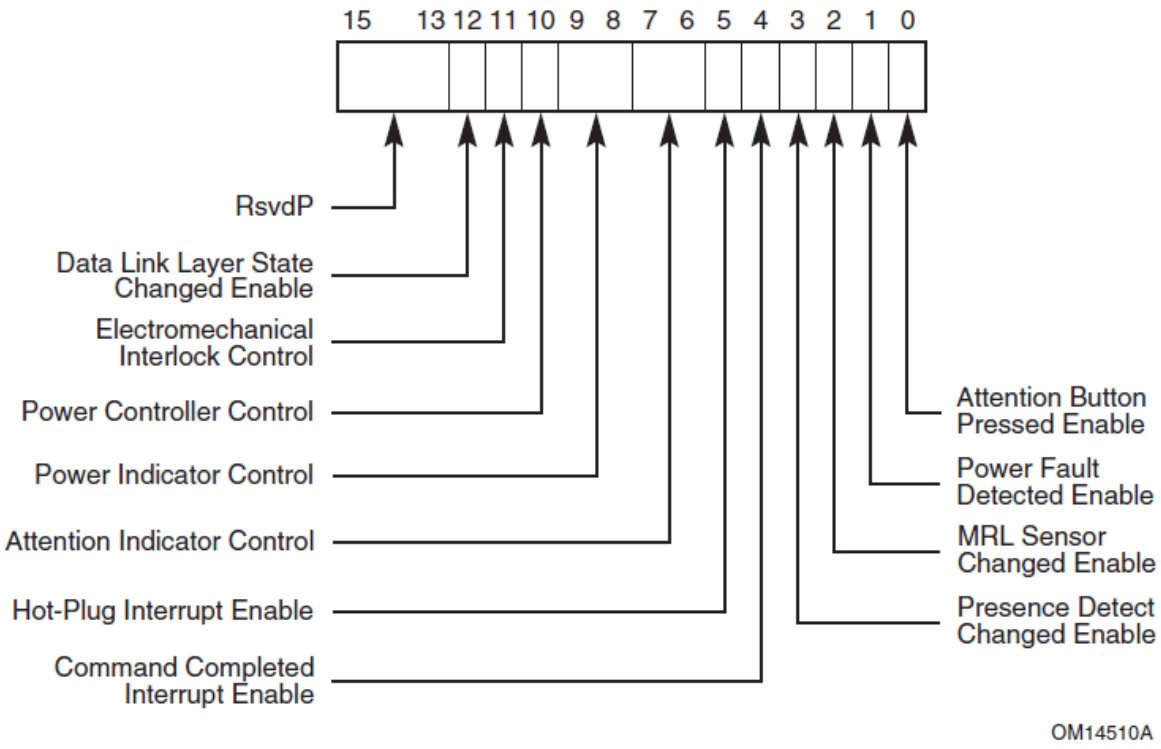


Figure 6-2: Slot Control Register (PCIe Figure 7-19)

Bit Field	Implement, Zero, One, Optional
Data Link Layer Changed Enable	Implement
Electromechanical Interlock Control	Zero
Power Controller Control	Implement
Power Indicator Control	Implement
Attention Indicator Control	Implement
Hot-Plug Interrupt Enable	Implement
Command Completed Interrupt Enable	Implement
Presence Detect Changed Enable	Implement
MRL Sensor Changed Enable	Zero
Power Fault Detected Enable	Implement
Attention Button Pressed Enable	Implement

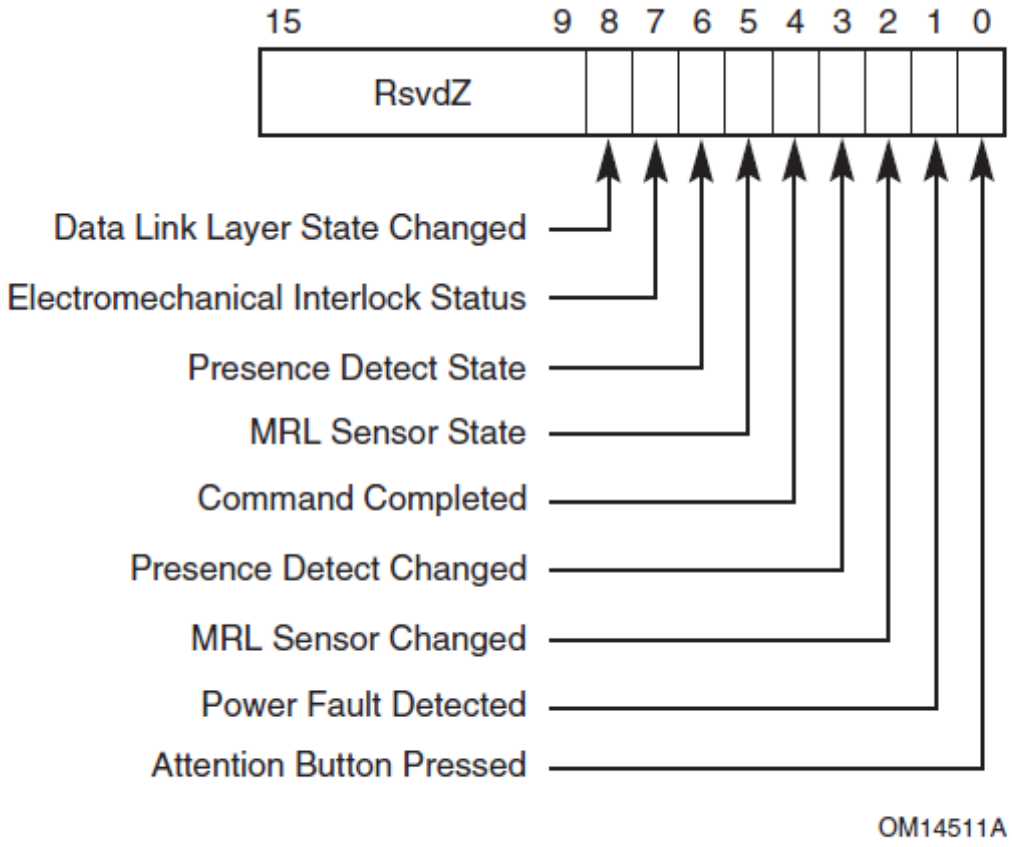


Figure 6-3: Slot Status Register (PCIe Figure 7-20)

Bit Field	Implement, Zero, One, Optional
Data Link Layer State Changed	Implement
Electromechanical Interlock Status	Zero
Presence Detect State	Implement
MRL Sensor State	Zero
Command Completed	Implement
Presence Detect Changed	Implement
MRL Sensor Changed	Zero
Power Fault Detected	Implement
Attention Button Pressed	Implement

6.2.2. Virtual Signals Interface Registers

Each virtual downstream Switch Port has a Virtual Signals Interface. Registers controlling this hardware are located using the MR-IOV capability and are visible only to MR-PCIM. They provide

the interface between MR-PCIM Hot-Plug software and Hot-Plug software running in the VH. This interface allows MR-PCIM to:

1. Provide configuration and status information the virtual slot registers
2. Push virtual buttons of the virtual Hot-Plug controller (i.e. change bits in the virtual Slot Status register)
3. Detect Indicator and Control changes made to the virtual Hot-Plug controller (i.e. detect certain changes to the virtual Slot Control and virtual Slot Capabilities registers)

The following register fields are proposed for this function:

Bit Field	Implement, Zero, One, Optional	Purpose
Physical Slot Number	Read / Write	1
Slot Power Limit Scale / Value	Read Only	3
Slot Implemented	Read / Write	1
Hot-Plug Capable	Read / Write	1
Hot-Plug Surprise	Read / Write	1
Data Link State	Read / Write	2
Power Controller State	Read Only	3
Power Controller State Changed	Read / Write One to Clear	3
Power Controller Present	Read / Write	1
Power Indicator State	Read Only	3
Power Indicator State Changed	Read / Write One to Clear	3
Attention Indicator State	Read Only	3
Attention Indicator State Changed	Read / Write One to Clear	3
Presence Detect State	Read / Write	2
Press Attention Button	Read Zero, Write One to Set	2
Signal Power Fault	Read Zero, Write One to Set	2

Note that these fields are exact duplicates of the corresponding Virtual Slot Register fields allowing some hardware simplification.

In addition to the above fields, any Indicator and Control changes by VH software (Purpose 4 above) can generate an interrupt to MR-PCIM. An interrupt enable and dispatch mechanism will be defined to allow MR-PCIM to efficiently manage these interrupts.

6.2.3. Physical Slot Registers

Each physical slot contains a set of registers for controlling the physical Hot Plug Controller.

When an MRA Switch is running in base PCI Express mode, this control occurs using the Slot Capability, Control, and Status registers associated with the Type 1 header of the physical downstream Switch Port.

When an MRA Switch is running in MR mode, this control uses the Slot Capability, Control and Status contained in the MR IOV capability block. For simplicity, register layout and functionality is identical as the registers described in *PCI Express Base Specification* sections 7.8.9, 7.8.10 and 7.8.11 (and excerpted above).

Note: These registers are controlled by MR-PCIM. They are only usable from the Management VH.

Note: The Management VH also contains Virtual Slot Registers and has an associated set of Virtual Signals Interface registers.

Note: An alternative approach would control the physical slot signals using the "Virtual" Slot Registers of the Management VH. This scheme has difficulties dealing with failover of the Management VH (i.e. what happens to the physical signals, interrupts, etc during a failover). Making the physical signals a distinct set of registers avoids these complexities.

6.2.4. Physical Signals Interface

This interface contains the usual base PCIe signals. The base specification does not fully specify these signals and neither does this specification.

6.3. MRA Device

A main use case for virtual Hot Plug is to support device migration. To support this use case, a mechanism to cleanse device state must be provided.

This can be accomplished in a variety of ways. One option is to issue Function Level Resets for the functions being reassigned. Another option is to turn off power to the affected functions by writing them to D3 state.

6.4. Non-MRA Components

PCIe components attached to a Switch in MR mode are assigned to a single VH. Hot plug operations are identical to the MR case except that only a single VH is involved.

The Switch Physical Slot Registers are used to control the physical signals of the slot. Virtual Slot registers provide a view of this information to the software running in the assigned VH. MR-PCIM software can use the virtual hot plug interface to support graceful migration of a collection of conventional components from one VH to another.

7

7. Power Management

[Primary authors: Steve Glaser / Chris Pettey]

MR systems continue to need power management capabilities. Two varieties of power management are involved:

- Virtual power management allows software running in a VH to believe that it has turned off power to one or more virtual functions.
- Actual power management allows MR-PCIM software to control device power.

7.1. Requirements

- The specification shall enable each VH to put its Functions into any supported D state
- The specification shall allow MR PCIM to enable/disable a physical device from actual power down state.
- The specification shall utilize PCIe mechanisms for changing the D state of a Function.

7.2. Virtual Power Management

MRA components contain power management capabilities in each VH. This provides PCIe compatibility to software running in each VH. Software funning in a VH must be able to place a function into D3 state and bring it back to D0. This function must behave like a real function experiencing the same transition (e.g. appropriate state must be cleared).

No actual power savings may occur as a result of placing components in virtual low power states.

Note: The Power Select, Power Data and Power Scale fields in the PCI Power Management capability and the entire PCI Power Budgeting capability remain optional.

7.2.1. Virtual Wake

For some device types, software can configure the device to wake up the system when certain events occur while the device is in low power state. Support for such devices in MR systems uses a virtual wake capability.

Virtual wakes in a shared portion of a hierarchy must turn into actual wakes in the non-shared portions of the same hierarchy.

A virtual wake will likely be represented as a PME message within the appropriate virtual hierarchy. MR Switches forward PME messages towards the root of the hierarchy.

7.3. Actual Power Management

When a component is placed into virtual low power state in all of its virtual hierarchies, the actual component may be moved to the same or higher low power state. The mechanism involved is vendor specific and can be a combination of software and hardware.

7.3.1. Actual Wake

When components are in actual low power state, virtual wake events need to trigger actual wake events. There are two cases to consider:

1. Virtual wake events waking up a non-virtual portion of a hierarchy.

 In this case, a virtual PME message needs to be forwarded out a *non-shared* link that is currently down. The MR Switch holds on to the PME, signals a wake on the appropriate link (Beacon or WAKE#) and forwards the PME message when the link comes up.

2. Virtual wake events waking up a portion of a hierarchy that involves some virtual components.

 In this case, a virtual PME message needs to be forwarded out a *shared* link that is currently down. This situation is an open area for workgroup investigation. The low power state behavior of MR IOV capabilities will be specified.

8

8. Congestion Management

Exceeding the bandwidth of a link or capacity of a buffer can lead to congestion in a topology. In a Multi-Root Topology, congestion may affect the performance of unrelated VHs and lead to Completion Timeout errors. This chapter defines mechanisms for detecting and controlling congestion in a Multi-Root Topology.

8.1. Overview

There are three possible causes of congestion in a PCIe topology.

- A fault in hardware or software configuration of a device in the topology.
- A static rate mismatch in the capacity of the path from a component injecting traffic into the topology (e.g., a Device) and the ultimate destination (e.g., a Root Port). Congestion due to a static rate mismatch would occur even if the topology were otherwise idle.
- Traffic merging of multiple flows, none of which individually suffer from a static rate mismatch, causing the capacity of an element in the topology to be exceeded

While the causes of congestion are the same in both single and multi-root PCIe topologies, it is desirable to provide the ability to manage, limit and contain the congestion caused by one VH on other VHs in the system.

8.2. Requirements and Goals

- The congestion management scheme shall allow systems to be constructed where a fault in one VH shall not result in Root Port Completion Timeouts in another VH.
- It is a goal of the congestion management scheme to allow systems to be constructed where a fault in one VH does not result in Device Completion Timeouts in another VH.
- In PCIe and Single Root Topologies, Virtual Channels (VCs) provide a means to support multiple independent logical data flows over a Link. The congestion management scheme shall allow systems to be constructed that preserve this behavior within a VH.
- It is a goal of the congestion management scheme to allow congestion to be managed due to traffic merging from different (VH, VC)s. This form of congestion is unique to MR topologies.
- It is not a goal of the congestion management scheme to address congestion due to traffic merging within a (VH, VC). This form of congestion is possible in both SR and MR topologies.

- It is not a goal of the congestion management scheme to address congestion due to a static rate mismatch. This form of congestion is possible in both SR and MR topologies.
- It is a goal of the congestion management scheme to allow systems to be constructed that ensure that congestion in one VH does not result in Device or Root Port Completion Timeouts in another VH.
- It is a goal of the congestion management scheme to allow systems to be constructed that support forward progress on a VH or groups of VHs when congestion exists on an unrelated VH.
- It is not a goal to require that all devices implement the congestion management mechanisms necessary to support complete isolation of congestion between VHs.
- It is a goal of the congestion management scheme to allow MR Devices to be constructed through incremental changes to SR Devices.
- It is a goal to standardize optional performance measurement capabilities in MRA Switches and MRA RCs to enable the detection of congestion.
- It is not a goal of the congestion management scheme to allow dynamic reallocation of resources to avoid congestion.

8.3. Virtual Links with Bypass Queues

A virtual Link (VL) provides the means to support multiple independent logical data flows over a single physical Multi-Root PCIe Link. A device may support one to eight VLs on a physical link. VLs play the same role in an MR topology as Virtual Channels (VCs) in a PCIe topology. VLs are associated with independent topology resources (e.g., queues and buffers) that are used to move information across an MR Enabled Link with independent flow control. Multi-Root Aware Components support one or more VHs. Each VH represents a PCIe hierarchy that may support one to eight VCs. Each (VH, VC) associated with an MRA component represents an independent logical data flow that in an MR topology must be mapped to a physical VL resource in order for data to be transferred across a link.

MR VL mechanisms are used to map (VH, VC)s onto VLs. (VH, VC)s may be mapped to VLs in a flexible manner (e.g., (VH 0, VC 0), (VH 1, VC 2), and (VH 2, VC 1) all map to VL 0) or on a VC consistent basis (e.g., (VH*any*, VC 0) all map to VL 0). Behavior is undefined when multiple VCs in the same VH are mapped to the same VL. While not a requirement, systems will generally map (VH, VC) data flows with similar QoS characteristics onto the same VL.

VLs are related with an MR Enabled Link and are not end-to-end. An MR topology may contain Links that implement a different number of VLs.

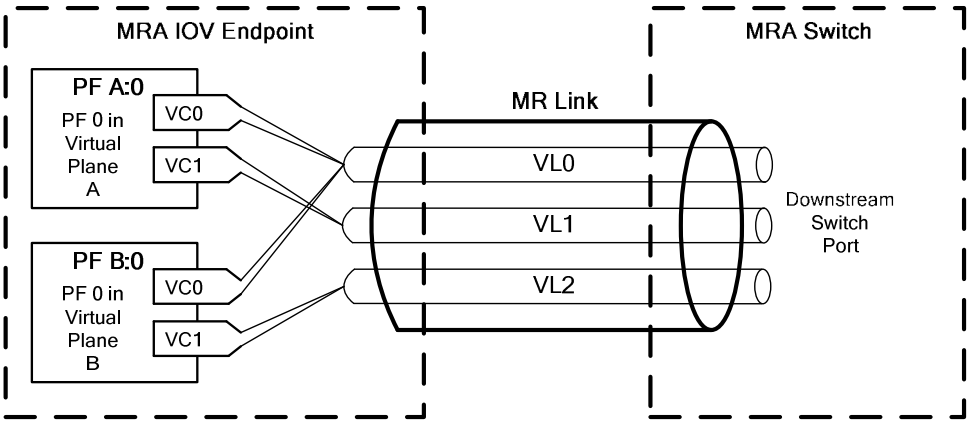


Figure 8-1: (VH, VC) to VL Mapping

A graphical example of (VH, VC) to VL mapping is illustrated in Figure 8-1, where an MRA IOV device connects to an MRA downstream Switch Port. In this example, the MRA IOV device implements two PFs, each with two virtual channels. One PF in this device has been assigned to VH A while the other to VH B. In this example VC 0 from both VHs has been mapped to VL 0 (i.e., (A, 0) and (B, 0) both map to VL 0) while VC 1 from VH A maps to VL 1 and VC 1 from VH B maps to VL 2.

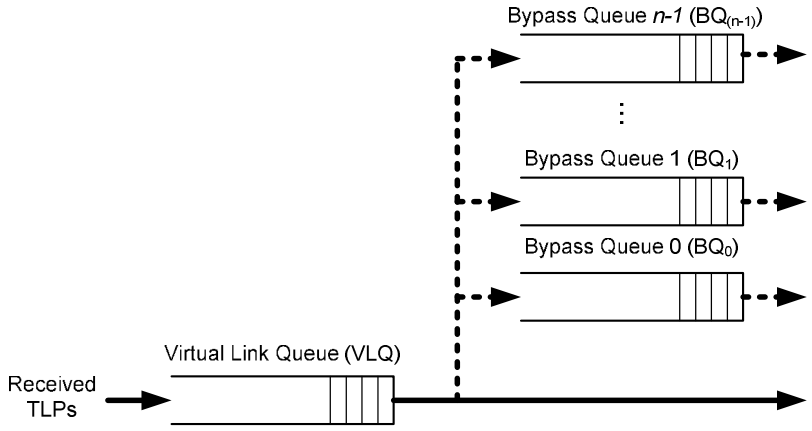


Figure 8-2: Logical Queuing Structure Associated with a VL Receiver

A logical queuing structure used to describe the behavior of a VL at a receiver is illustrated in Figure 8-2. This description does not imply or require a particular implementation and is used solely to clarify requirements. Associated with each VL at a receiver is a Virtual Link Queue (VLQ) and one or more optional Bypass Queues (BQ). PCIe ordering rules are maintained within a VH VL across both the VLQ and the associated BQ (if any). TLPs associated with a VL received on a Link are queued in the VLQ. In the absence of congestion, TLPs for non-congested (VH, VC)s are dequeued from the VLQ for receiver processing. When a TLP at the head of the VLQ experiences congestion, a free BQ is allocated to the (VH, VC) associated with the TLP, if one does not already exist, and the TLP is moved to the BQ. Subsequent TLPs from the VLQ are moved to the same BQ until the congestion condition clears and the (now empty) BQ is freed. This scheme allows TLPs associated

with a congested flow to be bypassed and isolates congestion to a (VH, VC). TLPs are dequeued from a BQ as flow control credits of the required type for the (VH, VC) associated with the BQ become available. An implementation must ensure that forward progress is guaranteed on all flows.

If at any point the number of congested (VH, VC)s associated with a VL exceeds the number of BQs implemented by a receiver for that VL, then TLPs in the VLQ experience congestion. Thus, congestion is managed within a VL at the receiver by dynamically mapping congested (VH, VC) flows onto BQs. The degree of congestion isolation between flows mapped to the same VL is dictated by the number of BQs implemented for that VL by the receiver. A receiver that implements only the required VLQ provides no congestion isolation between data flows mapped to the VL, while a receiver that implements *n* optional BQs provides complete congestion isolation for up to *n* flows.

8.3.1. (VH, VC) Mapping

Support for VLs beyond the default VL 0 in MRA components is optional.

MRA components that implement multiple VLs must implement a (VH, VC) to VL mapping capability associated with VHs supported by a Port. Components that implement VHs that support multiple VCs shall provide MR PCIM with the capability to disable VC resources.

8.3.2. Arbitration

MRA components that support multiple VHs require an arbitration mechanism associated with each supported VL at egress to determine the VH from which the next TLP will be transmitted on that VL. This arbitration is referred to as VH to VL arbitration.

MRA components that support multiple VLs require an arbitration mechanism at egress Ports to determine the VL from which the next TLP will be transmitted on the physical Link. This arbitration is referred to as VL to Link arbitration.

It is a goal of VH to VL arbitration and VL to Link arbitration to provide the following:

- Guaranteed forward progress on all supported data flows.
- Differentiated service characteristics for data flows with an MR topology.
- The ability to tune bandwidth and end-to-end latency between components in an MR topology.

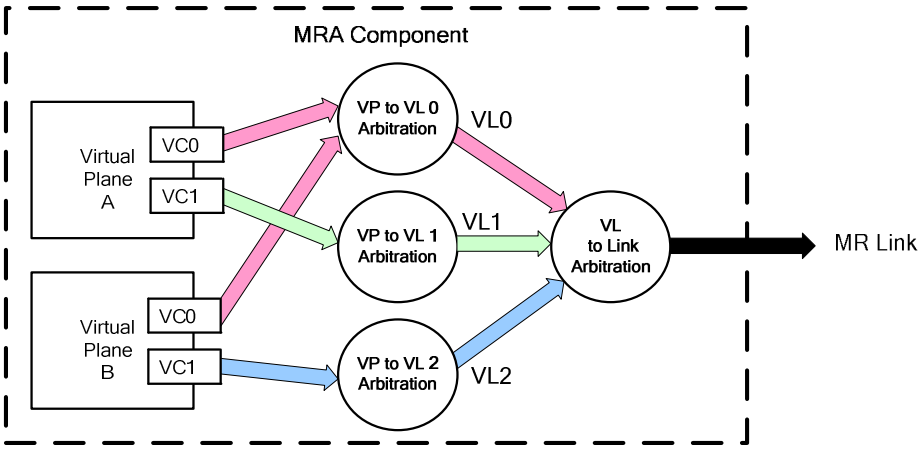


Figure 8-3: MRA Arbitration Model

Figure 8-3 illustrates both VH to VL and VL to Link arbitration associated with an egress Port of a MRA Enabled PCIe Component such as an MR RP, Switch, Bridge, or Device. The component in this example implements two VHs, with both VHs implementing two VCs. Flows (A, VC 0) and (B, VC 0) are mapped to VL 0 and require VH to VL arbitration to control the multiplexing of TLPs onto this VL. VLs one and two only have a single mapped flow and therefore require trivial arbitration. The physical link associated with the egress Port in this example implements three VLs. VL to Link arbitration is required to control the multiplexing of VLs onto the physical link.

8.3.2.1. VH to VL Arbitration

MRA components may implement an optional VH to VL arbitration capability associated with each VL supported by an egress Port.

VLs supported by an egress Port that do not implement the optional VH to VL arbitration capability must implement a hardwired-fixed arbitration scheme (e.g., round robin) that guarantees forward progress on all mapped VHs.

VLs supported by an egress Port that implement the optional VH to VL arbitration capability may support the following arbitration schemes:

- Hardwired-fixed arbitration (e.g., round robin).
- Programmable Weighted Round Robin arbitration.

8.3.2.2. VL to Link Arbitration

MRA components may implement an optional VL to Link arbitration capability associated with each egress Port.

Egress Ports that do not implement the optional VL to Link arbitration capability must implement a hardwired-fixed arbitration scheme (e.g., round robin) that guarantees forward progress on all active VLs.

Egress Ports that implement the optional VL to Link arbitration capability may support the following arbitration schemes:

- Strict priority
- Round robin
- Programmable Weighted Round Robin

8.4. MR Flow Control

Flow control information is exchanged between an MRA receiver and a transmitter associated with a Link using Flow Control Packets (FCPs), which are a type of DLLP.

- The length of these DLLPs shall be the same as in PCIe.
- As in PCIe, these DLLPs shall be protected by a CRC-16.
- The unit of Flow Control credit for header and data shall be the same as in PCIe.

Components must implement independent Flow Control for all VLs supported by the component.

8.4.1. SR to MR Enabled Link Transition

All links shall initially train using the PCIe protocol using only VC 0.

MR flow control information shall be initialized by MR PCIM in each component based on capabilities advertised by MRA components.

MR PCIM shall initiate a transition from PCIe protocol to MR PCIe protocol by writing to configuration registers in the upstream and downstream components.

It is a goal to allow MR PCIM to allow optional configuration of the number of supported VLs associated with a Link. MRA components shall advertise the following information to MR PCIM:

- TBD

8.4.2. FC Information Tracked by Transmitter

A transmitter shall track the following two quantities for each supported (VH, VC) and VL.

- CREDITS_CONSUMED
 - Computed in the same manner as in PCIe protocol.
- CREDIT_LIMIT
 - Configured by MR PCIM during MR Enabled Link initialization

The transmitter gating function must determine if sufficient credits are available to permit the transmission of a given TLP. If the Transmitter does not have enough credits to transmit the TLP, it

must block the transmission of the TLP. The Transmitter must following the same ordering and deadlock avoidance rules as specified in the PCIe protocol. TLPs mapped to different VLs have no ordering relationship, and must not block each other.

The transmitter gating function rules for (VH, VC)s and VLs are the same as that specified in the PCIe protocol.

The transmitter gating function for a TLP is said to pass if the transmitter gating function passes for both the (VH, VC) and VL associated with the TLP.

TLPs associated with different VHs represent different flows and have no ordering relationship. TLPs blocked due to failure of the (VH, VC) gating function should not block TLPs associated with other VHs mapped to the same VL.

8.4.3. FC Information Tracked by Receiver

A receiver shall track the following two quantities for each supported (VH, VC) and VL.

- CREDITS_ALLOCATED
 - For VLs this value is initially set to the value advertised by the component to MR PCIM.
 - For VLs this value is incremented as additional receiver buffer space becomes available in the VLQ. This may be due to the Receiver Transaction Layer making buffer space available due to processing of Received TLPs, or due to congested TLPs being moved to a BQ.
 - For (VH, VC)s this value is initially set to the value advertised by the component to MR PCIM or that configured by MR PCIM.
 - For (VH, VC)s this value is incremented as additional receiver buffer space becomes available in the VLQ or an allocated BQ due to processing of Received TLPs associated with the (VH, VC).
- CREDITS_RECEIVED
 - Computed in the same manner as in PCIe protocol.

It is a goal to reduce the amount of FCP traffic associated with inactive flows. After an MR UpdateFC has been sent TBD [proposed 4] times with the same value, subsequent MR UpdateFC FCPs must be scheduled for transmission once every TBD [proposed 100 ms] (-0%/+50%). To prevent false completion timeouts, it is recommended that the Completion Timeout Ranges Supported be set to range TBD or higher.

8.4.4. Virtual Hierarchy (VH) Reset

The resetting of a VH represents a logical event that does not affect the operation of a physical MR Enabled Link or flow control information tracked on the Link.

Once enabled, MR mode flow control information is tracked by Receivers and Transmitters for all configured VLs and (VH, VC)s. The resetting of a VH in no way affects this operation. A receiver is

still required to send flow updates for shared resources, even if a VH goes into reset and TLPs on that VH are flushed/purged.

See Section 2.4.3.3 Flow Control and Reset / DL_DOWN for additional details.

8.4.5. Enabling and Disabling of VCs

Once enabled, MR mode flow control information is tracked by Receivers and Transmitters for all configured VLs and (VH, VC)s. The enabling or disabling of a VC within a VH represents a logical event that does not affect the operation of a physical MR Enabled Link or flow control information tracked on the Link.

8.5. Performance Monitoring

Congestion in a Multi-Root Topology may result in traffic associated with one Virtual Hierarchy affecting the performance of an unrelated Virtual Hierarchy. This section defines a set of optional performance monitoring capabilities that may be used to diagnose, plan and tune the performance of a multi-root system.

The objective of standardizing performance monitoring capabilities is to allow Device vendor independent software to monitor performance and manage congestion. It is not a goal of these capabilities to monitor or count errors.

Performance monitoring is a capability associated with a Device Port. This capability is optional for all MRA component Ports but its implementation is strongly encouraged for all MRA Switch Ports.

8.5.1. Statistics Collections

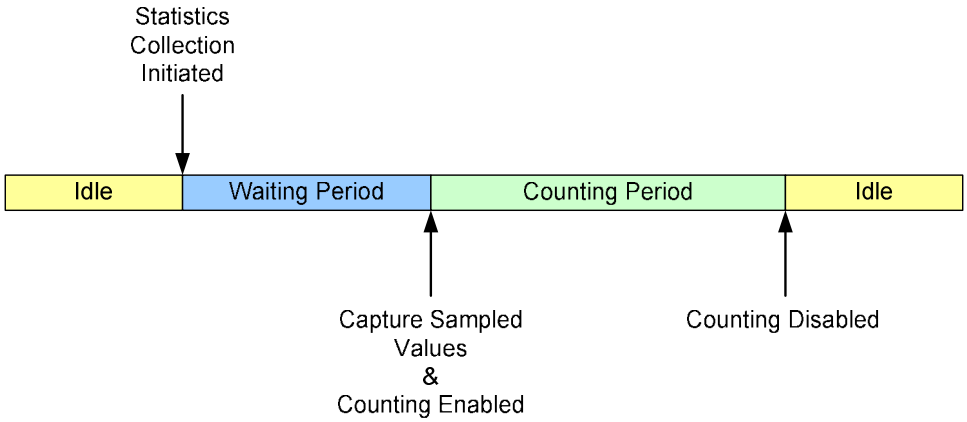


Figure 8-4: Statistics Collection Process

Statistics collection is illustrated in Figure 8-4 and refers to the process of collecting performance statistics associated with a Port.

Statistics collection may be initiated independently on a per Port basis by writing to a register field associated with a port or globally on all ports of component by writing to a register field associated with the entire component.

Port performance statistics consist of captured values and counter values. Captured values correspond to sampled port state while counter values correspond to the number of occurrences of a selected event over a sampling period.

Captured values are sampled and counter counting is enabled after a waiting period following initiation of statistics collection. The length of the waiting period is specified by a field 16-bit register field associated with the port. This field specifies the waiting period in units of microseconds.

Following the waiting period is the counting period over which selected event occurrences are counted. The length of the counting period is specified by a 24-bit register field associated with the Port. This field specifies the counting period in units of microseconds.

The state of a Port's statistics collection process may be determined by reading a status field associated with the Port.

8.5.2. Counter Values

A counter value corresponds to the value of a 32-bit saturating counter that counts the number of occurrences of a selected event during the counting period. Associated with a counter is a select field that specifies the event being counted. Both the select field and the 32-bit counter are contained in register field associated with the Port.

If a port implements the performance monitoring capability, then it must implement at least two counters.

Each implemented counter must be capable of counting the following events associated with a port.

- Number of transmitted TLPs3
- Number of received TLPs4
- Number of transmitted TLP DWORDs (from STP to END)
- Number of received TLP DWORDs (from STP to END)
- Number of transmitted idle Symbols
- Number of received idle Symbols

Each implemented counter may optionally support the counting of the following events. Each of these events may correspond to any TLP associated with the port or be further filtered based on VL, VHN, and TLP type (i.e., Posted, Non-Posted or Completion)

- Events specified above as required with VL, VHN or TLP type filtering

[3] A transmitted TLP in this section refers to non-nullified TLPs

[4] A received TLP in this section refers to non-nullified TLPs that pass Data Link Layer error checking

- Number of Symbol Times a TLP is blocked from transmission due to lack of VL flow control credits
- Number of Symbol Times a TLP is blocked from transmission due to lack of (VHN, VL) flow control credits
- Number of Symbol Times a TLP is blocked from transmission due to lack of VL or (VHN, VL) flow control credits
- Number of transmitted DLLPs of any type
- Number of received DLLPs of any type

A range of counter selects shall be reserved for counting of vendor specific events.

8.5.3. Sampled Values

A sampled value is a value whose state is captured at the start of the counting period. Associated with a sampled value is a select field that specifies the value to be sampled. Both the select field and the sampled value are contained in registers associated with the Port.

A port that implements performance monitoring is not required to implement sampled values.

Each implemented sampled value may optionally be configured to capture the following state.

- The number of available transmit credits associated with a VL computed as

 $$(\text{CREDIT_LIMIT} - \text{CREDITS_CONSUMED}) \bmod 2^{[\text{Field Size}]}$$

- The number of available transmit credits associated with a (VHN, VL) computed as

 $$(\text{CREDIT_LIMIT} - \text{CREDITS_CONSUMED}) \bmod 2^{[\text{Field Size}]}$$

- The number of available receive credits associated with a VL computed as

 $$(\text{CREDITS_ALLOCATED} - \text{CREDITS_RECEIVED}) \bmod 2^{[\text{Field Size}]}$$

- The number of available receive credits associated with a (VHN, VL) computed as

 $$(\text{CREDITS_ALLOCATED} - \text{CREDITS_RECEIVED}) \bmod 2^{[\text{Field Size}]}$$

A range of sampled value selects shall be reserved for sampling of vendor specific values.